MEDiSN: Medical Emergency Detection in Sensor Networks

JEONGGIL KO, JONG HYUN LIM, YIN CHEN, RĂZVAN MUSĂLOIU-E., ANDREAS TERZIS and GERALD M. MASSON
Johns Hopkins University
TIA GAO and WALT DESTLER
Aid Networks
and
LEO SELAVO
University of Latvia
and
RICHARD P. DUTTON
University of Maryland Medical Center

Staff shortages and an increasingly aging population are straining the ability of emergency departments to provide high quality care. At the same time, there is a growing concern about the hospitals' ability to provide effective care during disaster events. For these reasons, tools that automate patient monitoring have the potential to greatly improve efficiency and quality of health care. Towards this goal, we have developed *MEDiSN*, a wireless sensor network for monitoring patients' physiological data in hospitals and during disaster events. MEDiSN comprises *Physiological Monitors* (PMs) which are custom-built, patient-worn motes that sample, encrypt, and sign physiological data and *Relay Points* (RPs) that self-organize into a multi-hop wireless backbone for carrying physiological data. Moreover, MEDiSN includes a back-end server that persistently stores medical data and presents them to authenticated GUI clients. The combination of MEDiSN's two-tier architecture and optimized rate control protocols allows it to address the compound challenge of reliably delivering large volumes of data while meeting the application's QoS requirements. Results from extensive simulations, testbed experiments, and multiple pilot hospital deployments show that MEDiSN can scale from tens to at least five hundred PMs, effectively protect application packets from congestive and corruptive losses, and deliver medically actionable data.

Categories and Subject Descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.3 [**Special-Purpose and Application-Based Systems**]: Real-time and Embedded Systems; J.3 [**Life and Medical Sciences**]: Medical Information Systems

Authors' addresses: JeongGil Ko, Jong Hyun Lim, Yin Chen, Răzvan Musăloiu-E., Andreas Terzis and Gerald M. Masson, Department of Computer Science, Johns Hopkins University, 3400 N.Charles St., Baltimore, MD 21209; Tia Gao and Walt Destler, Aid Networks, 155 Gibbs St. Rockville, MD 20850; Leo Selavo, Department of Computer Science, University of Latvia, Raina Blvd. 19, Riga, LV-1586, Latvia; Richard P. Dutton, 22 S. Greene Street, Baltimore, MD 21201; email: jgko@cs.jhu.edu

General Terms: Design, Experimentation, Performance
Additional Key Words and Phrases: Medical Sensor Networks, Wireless Physiological Monitoring

1. INTRODUCTION

The capacity of the US healthcare system has not kept pace with the growing demand for emergency and trauma care. A majority of urban Emergency Departments (EDs) today operate at or over capacity [American Hospital Association 2005], patients are left under-monitored [Coalition for American Trauma Care 2006], and fatalities due to the lack of monitoring have already occurred [CNN 2006; MSNBC News 2006; Stark 2006]. Furthermore, when disasters occur, the surge of patients can quickly overwhelm the already overcrowded care facilities. Thereby, an approach that automates the patient monitoring process has the potential to greatly improve the efficiency of patient flow, increase the volume of patients treated and discharged, and improve the quality of care both on a daily basis and during disasters.

With this need in mind, we present *MEDiSN*, a wireless sensor network for automating the process of patient monitoring in hospitals and disaster scenes. MEDiSN consists of multiple *Physiological Monitors (PMs)* which are battery-powered motes equipped with sensors for measuring patients' physiological data (e.g., blood oxygenation level, pulse rate, electrocardiogram (ECG), etc.). PMs temporarily store collected measurements and transmit them after encrypting and signing them. Unlike previous systems (e.g., CodeBlue [Malan et al. 2004]) in which PMs also relay data, the MEDiSN architecture incorporates distinct *Relay Points (RPs)* which self organize into bidirectional wireless trees connecting the PMs to one or more *Gateways*. Traffic flowing in both directions is protected using hop-by-hop retransmissions that counter the effects of packet collisions and corruptions.

The division of functionality between acquiring and relaying data enables PMs to achieve consistent, predictable behavior and low energy consumption, while allowing us to engineer the system to provide superior end-to-end service. Specifically, while PMs can be mobile, the RPs have pre-determined, fixed positions that allow us to provision a high-quality wireless backbone. Furthermore, because PMs are not responsible for relaying traffic, they can aggressively duty cycle their radios to reduce energy consumption. On the other hand, duty cycling is not an option for RPs as they are always busy forwarding packets. Nonetheless, RPs will use the electricity grid in hospital deployments, while in disaster events batteries can power RPs for multiple days.

We evaluate MEDiSN's performance through extensive experiments in simulated environments, two indoor testbeds, and multiple pilot hospital deployments. From a networking perspective, we show that dynamically adjusting the maximum number of retransmissions the RPs attempt and computing the optimal inter-packet intervals can increase the system's capacity by ∼50% and reduce end-to-end delay by threefold. Moreover, properly engineering the RP backbone can increase the packet delivery ratio up to threefold as well. Additionally, we show that aggregation, in which RPs gather multiple small PM messages to maximum length packets,

can increase the total number of PMs supported by 30%, while achieving $\sim 90\%$ delivery ratio. Results from indoor testbeds indicate that the performance of the actual system accurately matches the simulation results. Using testbed results we also show that MEDiSN outperforms CodeBlue both in terms of delivery ratio for the same number of PMs and in terms of the maximum number of supported PMs. Finally, experimental results suggest that the two-tier routing hierarchy combined with the RP selection scheme allows MEDiSN to perform well even with multiple mobile PMs.

We have deployed MEDiSN in multiple IRB-approved clinical studies at the trauma center of the University of Maryland Medical Center and the Johns Hopkins Hospital Emergency Department. These deployments proved that MEDiSN can effectively retrieve data from multiple mobile PMs despite RF-challenging environments with multiple occlusions and considerable interference. Moreover, these studies showed that MEDiSN delivers vital signs measurements that are statistically identical to those generated by (wired) patient monitors commonly used in clinical practice.

Our Contributions: (1) We have conducted extensive interviews with hospital personnel and first responders and assembled a comprehensive list of requirements for sensor networks used in mass casualties and clinical environments. **(2)** To meet these requirements we present a unified, hierarchical network architecture that can seamlessly support high data rate senders (e.g., PMs transmitting ECG data) and large numbers of low data rate senders. MEDiSN achieves this goal through a combination of provisioning (i.e., appropriately constructing the RP backbone), application intelligence, and dynamic network controls. **(3)** Moreover, we show that the division of responsibilities between PMs and RPs, enables PMs to be simpler and more energy efficient, while allowing for mobility with marginal loss in performance. **(4)** Our experience from piloting MEDiSN in urban hospitals shows that wireless sensor networks can support mission-critical applications in complex clinical settings.

This paper has six sections. The section that follows presents the requirements of patient monitoring applications, while Section 3 outlines MEDiSN's overall architecture and describes its components. We devote Section 4 to the system's evaluation and present related work in Section 5. Finally, we conclude in Section 6 with a summary.

2. BACKGROUND AND MOTIVATION

Given the U.S. demographic trends (aging population, increasing prevalence of chronic diseases), nursing staff shortages [American Association of Colleges of Nursing 2004], and decreasing hospital capacities [The Center for Disease Control 2005], it is no surprise that the U.S. healthcare system is facing immense challenges on a daily basis. Moreover, there is growing concern about the hospitals' ability to provide effective care during disasters, when the surge of patients can be overwhelming.

The use of inefficient tools in the patient care process is one of the root causes behind overcrowded hospitals. Nurses conduct "rounds" by manually measuring physiological data and documenting assessments on paper. Thereby, tools that automate the patient monitoring process have the potential to greatly improve the

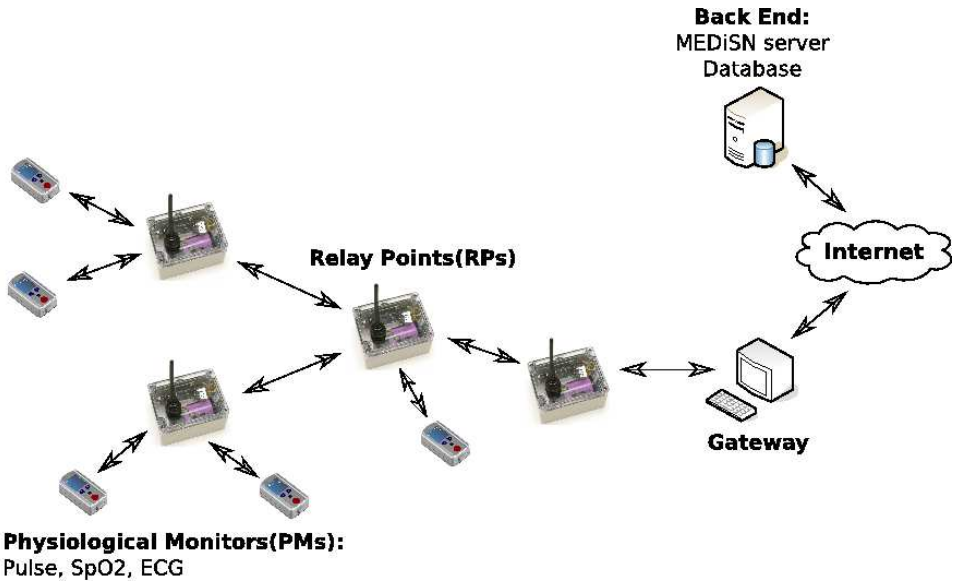

Fig. 1. The MEDiSN architecture, including a number of Physiological Monitors (PMs) collecting patients' physiological data, Relay Points (RPs) that forward the data, and a gateway which collects all the measurements and sends management commands. The back-end server hosts the GUI, persistently stores the patients' data, and can support multiple gateways.

quality of health care both on a daily basis and during disasters. We exemplify these benefits through the following two scenarios.

Mass Casualty Events: Monitoring patients becomes particularly challenging during disasters or surge scenarios, when existing infrastructure may be unavailable. In these cases, EMS (Emergency Medical Services) personnel should be able to quickly deploy physiological monitors that continuously relay the patients' physiological data and triage levels to one or more incident commanders on the scene.

Hospital Use: Patients commonly wait many hours in the waiting rooms of emergency departments after their condition has been initially assessed. Their condition, however, during that time may deteriorate and medical personnel cannot detect these potentially life-threatening events.

Wireless monitors, often called "telemetry monitors", available on the market today require costly installation of network infrastructure, are expensive, cannot be configured to fit users' workflow needs, and cannot scale up during disasters. These issues have long posed barriers to the mass adoption of wireless patient monitoring systems. In order to better understand the limitations of existing technology and the end-users' requirements we conducted extensive interviews with hospital personnel and first responders from five academic hospitals and two community hospitals. The key requirements generated from this process are as follows:

Scalability. The system must be able to support hundreds of patients and degrade

gracefully as the number of monitored patients increases.

Geographic reach. The network should be able to span wide geographic areas because in the event of a mass casualty disaster patients will overflow on the indoor and outdoor areas surrounding the emergency department. Furthermore, the network should be easy to deploy and not require any external infrastructure.

Traffic direction and types. In addition to patients' physiological data, the system must deliver critical messages, such as panic alerts from the patients as well as text alerts and management data from the gateway to individual patients.

QoS requirements. Physiological data should be delivered with an end-to-end latency of less than five seconds. Furthermore, alerts from patients should be delivered with higher probability than physiological data.

Security. According to U.S. law, medical devices must meet the privacy requirements of the 1996 Health Insurance Portability and Accountability Act (HIPAA). In order to meet this requirement the system must never broadcast identifiable patient data and guarantee the authenticity of the data delivered.

Mobility Support. For wireless medical sensing applications to be widely used, patient mobility should not significantly disrupt network performance.

Physiological Monitor Characteristics. The physiological monitor should be lightweight, fool-proof to use, and extensible, allowing the connection of additional sensors. Moreover, the monitor including its battery should be enclosed in a case that meets the requirements of medical devices that come in contact with patients. Finally, the monitor should have a five-day battery life which is the typical time that patients stay in US hospitals. From a software perspective, all of the monitor's parameters should be remotely configurable.

Presentation. The data that physiological monitors collect should be presented through a GUI that is intuitive and easy to use, as providers are often overworked and have little tolerance for tools that decrease their productivity.

3. ARCHITECTURE

Figure 1 provides a pictorial overview of MEDiSN's architecture. The figure includes a number of *Physiological Monitors (PMs)* – motes that record patients' physiological data. Each PM samples its onboard sensors at a configurable frequency, encrypts and signs[1] the resulting physiological data, and transmits them to one of the network's *Relay Points* (RPs). The RPs forward the PMs' data to a *Gateway* as well as management data from the gateway to one or more PMs. To support this two-way traffic, the RPs self-configure into a routing tree with the PMs at its leaves. While the RPs are stationary, PMs can be mobile and periodically select the best RP to forward their data to.

The paragraphs that follow detail each of MEDiSN's components.

[1]MEDiSN handles medical data which, by law, must remain private. In order to meet these requirements we designed and implemented a security software layer in TinyOS 2.x that exposes the security primitives the CC2420 radio offers [Hopkins interNetworking Research Group (HiNRG) 2008].

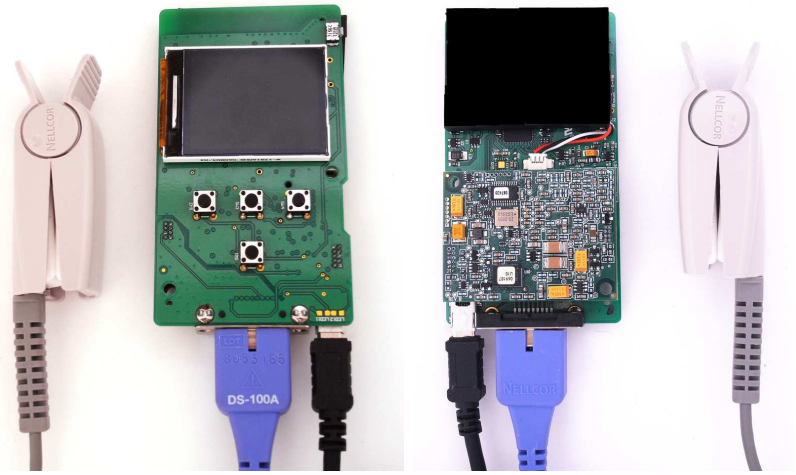

Fig. 2. One of the physiological monitors that MEDiSN supports. This Tmote Mini based monitor uses an 128x160 LCD screen, four buttons, and five LEDs for its user interface. Also shown, is a pulse oximetry probe connected via a DB9 connector to the mote's sensor board and a mini USB port used for serial communications and recharging the mote's battery.

3.1 Hardware

Drawing upon experience from developing multiple iterations of mote-based medical monitors [Selavo et al. 2006] and pilots in hospital and emergency response settings, we designed an extensible hardware platform that provides multiple patient monitoring capabilities (see Figure 2). The device is packaged in a water-resistant enclosure which includes a display, four buttons, and a buzzer for alerts.

The mote's processing core is the Sentilla Tmote Mini that combines a Texas Instruments MSP430 microcontroller with a TI/Chipcon CC2420 IEEE 802.15.4 radio in a mini-SDIO form factor [MoteIV Corporation 2007]. The mote is powered by a rechargeable 1,200 mAh 3.7 V Li-Ion battery. The battery voltage is up-converted to 5 V to power the sensors and down-converted to 3.3 V to power the Tmote Mini. The mote's actuators (vibrator, LEDs) are integrated via a standard I2C data bus. The PM also includes an LCD display and its sensing capability is provided by a Nellcor OEM controller [Nellcor Puritan Bennet Inc. 2006] that connects via a DB9 connector to Nellcor OxiMax sensor probes. An assortment of disposable and reusable sensor probes are available, including finger pulse clips for adults and foot wraps for pediatric patients. This Nellcor unit, optimized to be small in size, contains signal processing capabilities that allow consistent pulse oximetry (SpO_2) and pulse rate measurements even during challenging monitoring conditions caused by low perfusion, patient motion, and other forms of signal interference. The Nellcor controller connects to the microcontroller's control ports and its USART through a level shifting circuit. Finally, the mote includes a mini-USB port for reprogramming and recharging the battery.

We sought feedback from the user community on the design of the PM's user

interface. Decisions on the placement, size, and design of all user interface components carefully considered the monitor's diverse user requirements. We validated the monitor's design through multiple usability walk-throughs and pilots in hospitals and with first responders. All UI components, including the LCD display, four buttons, five LEDs, and a vibrator, are designed to be remotely configurable, allowing the hardware to dynamically adjust to the users' evolving needs. The 1.8" LCD screen, with a resolution of 128 $\times$ 160 pixels and 65K colors, can display physiological data and messages using multiple-sized fonts and custom icons.

While the mote shown in Figure 2 measures a patient's pulse rate and blood oxygen levels, MEDiSN also supports motes with different sensor suites, including a two-lead electrocardiogram (ECG) sensor [Fulford-Jones et al. 2004].

3.2 Routing

Physiological Data Traffic. The RPs form a routing tree to forward their measurements to the gateway. Rather than designing yet another tree routing protocol, we decided to base ours on the Collection Tree Protocol (CTP), included in TinyOS 2.x [Gnawali et al. 2009]. We use CTP because tree routing protocols are a maturing area of research and there is evidence that CTP selects high-quality end-to-end paths compared to other tree routing protocols [Fonseca et al. 2007].

At the same time, CTP has been primarily used by low data rate applications, while MEDiSN networks can potentially produce large amounts of data. Fortunately, CTP's modular architecture enables us to modify only the components necessary to support MEDiSN. Specifically, CTP consists of two major components, the *routing engine*, responsible for discovering and maintaining high-quality, multihop paths to the root of the tree, and the *forwarding engine* which controls the transmission of a node's packets to its parent. The forwarding engine determines the interval between successive packet transmissions as well as the maximum number of retransmissions. Moreover, the forwarding engine provides the ability for an application to intercept the messages relayed by the RP.

While CTP's routing engine successfully chooses good links in clinical environments [Ko et al. 2009], results in Section 4.1 indicate that CTP's forwarding engine is inefficient in loaded networks, leading to high loss rates. This is due to two factors: 1) the head of an RP's packet queue can be blocked for a long time due to the large number of retransmissions that CTP uses, thereby leading to queue overruns, and 2) the inter-packet interval (IPI) used by the forwarding engine to pace the transmission of consecutive packets is overly long. We modified CTP's forwarding engine in two ways to address these problems.

First, we implemented an algorithm that dynamically adjusts the maximum number of retransmissions as a function of the RP's queue size. Specifically, when the queue size grows above a threshold we reduce the maximum number of retransmissions by half. On the other hand, if the queue is below another (lower) threshold and packets exhaust their retransmission attempts, we increase the maximum number of retransmissions by one. This mechanism ensures that when the queue is long, the RP makes a "best effort" attempt to transmit the packets and discards them before they lead to multiple tail drops. This behavior is important in applications such as MEDiSN in which the system should provide the most up-to-date informa-

tion with minimal latency. If all packets cannot reach the gateway due to increasing congestion levels, more recent packets should have priority over older data.

The second modification dynamically adjusts the IPI that CTP uses. By default, a CTP sender waits for a random interval between 16 and 31 msec after a successful transmission before it attempts to transmit the next packet. While acceptable for low-rate networks, these conservative timers create considerable congestion in MEDiSN. Instead, each RP measures the delay between the time that a packet is ready to be transmitted (i.e., it reaches the head of the transmission queue) and the time that the RP receives an acknowledgement for that packet's successful transmission. This delay, T_p, which is in effect the amount of time that a node has to wait before it successfully transmits the packet, reflects the contention level in the RP's one-hop neighborhood. To see why this is the case, consider that when multiple nodes contend for the medium, they increase the probability of a node having to perform more CSMA backoffs thereby increasing T_p. Setting the IPI shorter than T_p will only increase the contention level at the RP's one-hop neighborhood without increasing its throughput. Alternatively, RPs maintain an exponentially weighted moving average (EWMA, $\alpha = 0.5$) of all the previous T_p values and set the IPI to this value. We present the performance benefits from this scheme in Section 4.1.

PMs, on the other hand, do not use CTP. Doing so would potentially involve PMs acting as relays and would create an inordinate number of tree reconfigurations as PMs can be mobile. Instead, a PM sends its data to the RP that shares the best link with that PM. We describe the RP selection mechanism in Section 3.3. A PM will retransmit a packet for a maximum number of times or until it receives an acknowledgment from the RP it is associated to. Also, to further increase the probability of delivering alert messages that have higher priority to the gateway, the PM includes these alerts to every packet that it transmits while the alert is active. A byproduct of not using CTP on the PMs is that PMs can turn their radio off after receiving an acknowledgement thereby conserving energy. Upon receiving a PM's packet, the RP adds a CTP header to it and adds it to its CTP queue from which it will be eventually transmitted.

Downstream Traffic. RPs also route downstream messages from the gateway to individual PMs. To do so, each RP periodically generates a Patient Information Packet (PIP) and forwards it to the gateway. The PIP includes all the PMs which connect directly to that RP, as well as the RP's identity. RPs that receive a PIP, append their ID before forwarding it toward the gateway. Then, when the PIP eventually arrives at the gateway, it contains a list of RPs that can be used to reach those PMs.

When the gateway needs to send a downstream message to one of those PMs, it generates a route to the PM by reversing the list of RPs contained in the PIP message. Downstream messages use the same hop-by-hop retransmission mechanism to increase the probability of delivering the gateway's messages to the PMs. Moreover, because PMs duty cycle their radios, the last RP buffers the downstream message until the PM wakes up to send its next batch of physiological data. Finally, the PM piggybacks an acknowledgment for each downstream message that it receives on the subsequent packet it transmits to the gateway.

We note that this strategy also works with mobile PMs. As a PM moves away from one RP and associates itself with another, the new RP will generate a PIP that allows that gateway to update its downstream route information. Downstream messages that were potentially mis-delivered during this 'hand-off' period will be retransmitted by the gateway after their acknowledgment timers expire.

3.3 Relay Point Selection

The simplest way for a PM to deliver its data to an RP is to use broadcast. However, doing so might lead to multiple RPs receiving and forwarding the same packet towards the gateway. To prevent this inefficiency, each PM selects one RP and unicasts its data to it. To do so, each RP periodically broadcasts beacons to notify PMs of its existence. In turn, PMs use these beacons to update their lists of reachable RPs and the quality of the links to them. Specifically, each PM maintains a moving average of the Link Quality Indicator (LQI) values it calculates from incoming beacons. By selecting links with persistently high LQI values, PMs can ensure that the links have low loss and are symmetric [Lin et al. 2006].

When a PM joins the network, its list of reachable RPs is empty and so it initially broadcasts its measurements. However, after the PM listens to the RPs' beacons, it switches to unicast mode. Moreover, at this point the PM starts duty cycling its radio to conserve energy. The PM continues to use the same RP until it can no longer reach it (i.e., it fails to receive any acknowledgments from that RP). This can happen either because the PM has moved out of the RP's communication range or because the RP has failed. At that point, the PM keeps its radio on to receive new beacons. In the meantime, it selects the next best RP or reverts to broadcasts if the list is empty.

RPs which serve more PMs than they can support, can use this selection mechanism to divert some of the PMs they serve to other RPs. To do so, they can temporarily stop broadcasting beacons or stop acknowledging (some) PM packets, at which point these PMs will switch to alternate RPs. In Section 4.1 we evaluate the performance benefits this RP selection mechanism provides.

3.4 Back-end Server

The gateway shown in Figure 1 collects data from the PMs and forwards them to a back-end server. This server acts as MEDiSN's dissemination layer, receiving data from one or more gateways and disseminating them to multiple authenticated end-users.

The back-end server we designed uses an N-tier architecture, exposing a set of SOAP and REST-based web services which allow authenticated clients to access and control the sensor network(s) the back-end server manages. Some of the exposed services include sensor history retrieval, sensor reconfiguration, user authentication, and alarm monitoring and generation. The server itself is designed to be lightweight enough to execute on consumer laptops and to be rapidly deployable in a disaster scenario where time is of the essence.

The high aggregate data rate of incoming sensor streams and the multiple potential sources of failure (e.g., congestion at the server, the clients, or the network itself) introduce significant challenges to the design of a reliable server architecture. We selected a J2EE-based Message Oriented Middleware (MOM) architecture to

meet these challenges. Using MOM, a JMS-based message queuing system running on the gateway and the back-end server is responsible for storing, routing, and retransmitting messages until they are successfully received by the server. JMS is a widely-used standard which contains a variety of features useful to medical sensor network applications including built-in security, ability to broadcast messages to multiple subscribers, and data buffering when connectivity to the server is temporarily interrupted.

The server itself is divided into four layers: **(1)** The data layer which persistently stores sensor data to a database. **(2)** The messaging layer for transmitting data to and from gateways. **(3)** The business logic layer that processes sensor data as they arrive from the messaging layer. **(4)** The web services layer for exposing sensor data to GUI clients. Furthermore, a software cache, internal to the server, allows multiple clients to frequently poll the server for the latest alerts and sensor data. In this way, GUIs making web service requests over HTTP can receive data at rates comparable to those provided by push-based, stateful connections, while benefiting from the simplicity that stateless connections offer.

Our preliminary tests show that the server can support as many as 200 patients. To support these clients, the server consumed $\sim 40\%$ CPU time on an PC equipped with a 3 GHz Pentium D processor and 1 GB of RAM.

4. EVALUATION

We evaluate MEDiSN's performance through multiple mechanisms. First, we use extensive simulations to evaluate the performance of MEDiSN's networking layer under a wide variety of scenarios. Although we realize that the simulations do not fully capture reality, we use them to explore in a principled way the effect of different parameters (e.g., network size and configuration) on the system's performance. Doing so in actual deployments would be infeasible since we cannot control the environment. Nevertheless, we validate some of the simulation results with a prototype implementation deployed in two indoor testbeds and multiple pilot hospital deployments. Finally, we measure the PM's energy consumption under various configurations and different external sensors.

4.1 Routing

4.1.1 *TOSSIM Simulations.* We use the TOSSIM TinyOS simulator [Lee et al. 2007] to test MEDiSN's performance bounds and to investigate how different network configurations and conditions can influence its behavior.

To test MEDiSN with ECG samples we implement a simple lossless mechanism for ECG compression. We do this to reduce the amount of data that PMs transmit thereby reducing energy consumption and network load. Our ECG compression is based on the well-known adaptive delta coding approach used in signal processing [Abate 1967] and is similar to lossless compression mechanisms for ECG data in the literature [Jafari et al. 2006; Nygaard and Haugland 1998]. By detecting the R-wave (slope with highest and lowest peak in an ECG waveform) we compress the 375 bytes of ECG data (our ECG sensor generates samples at a rate of 250 Hz with each data sample being a 12-bit digitized number) to 172 bytes per second. Because even this compressed packet is larger than the maximum IEEE 802.15.4 payload, each PM transmits one packet containing 86 bytes of ECG data every

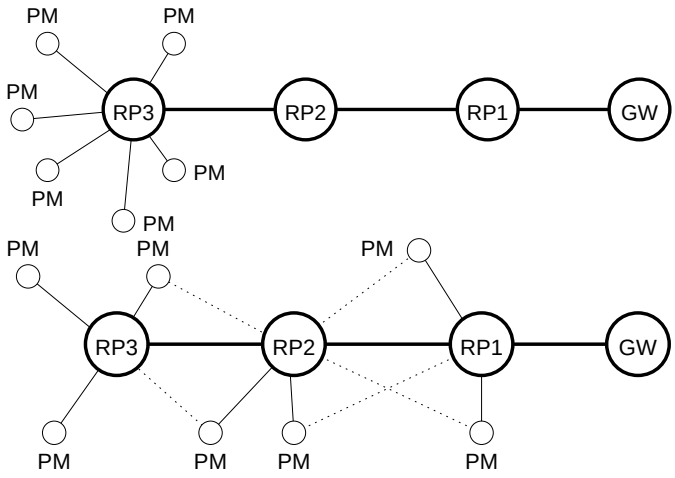

Fig. 3. Representative simulation topologies: The mushroom topology is shown at the top where all PMs transmit to the RP farthest away from the gateway. In the line topology shown at the bottom, PMs are distributed across different RPs. Note that some PMs can connect to multiple RPs in the line topology.

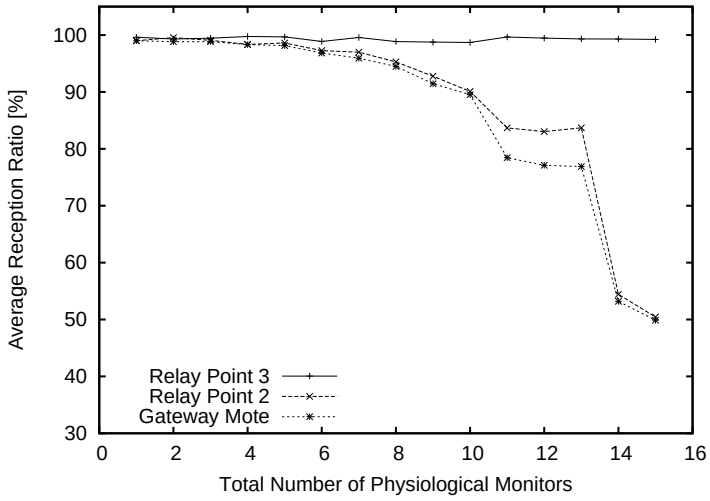

Fig. 4. Average reception ratio for the mushroom topology under perfect channel conditions. Different lines represent the percentage of packets received by different nodes in the topology.

500 msec. The total packet size including the CTP, TinyOS, 802.15.4 headers is 104 bytes. Furthermore, the PM transmits another packet of the same size every second containing pulse oximetry values, pulse rate and other control data such as LCD information and battery status. In summary, a node transmits three 104 byte packets per second: two ECG data packets and one packet contains pulse oximetry, pulse rate and control data.

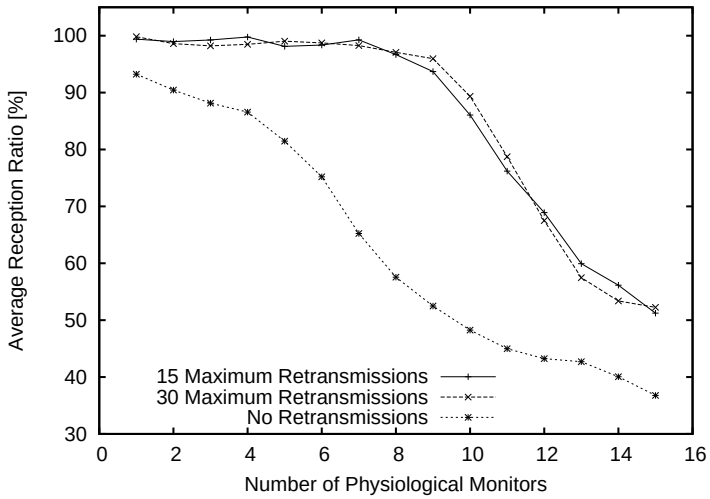

Fig. 5. Average reception ratio at the gateway for the mushroom topology and a noisy channel. Different lines correspond to different maximum number of retransmissions at the RPs.

4.1.2 *Mushroom Topology.* We present MEDiSN's capacity in terms of the maximum number of PMs it can support while providing good service (i.e., low loss and low delay). Moreover, we identify the root factors that control performance.

To do so, we start with a simple mushroom topology (Figure 3 presents our topology terminology), gradually increasing the number of PMs connected to RP 3. The maximum number of retransmissions was set to 15 for both the RPs and the PMs. As Figure 4 indicates, most packets are successfully received by the first RP (i.e., RP 3). On the other hand, many of these packets do not arrive to RP 2. Most of these losses are due to queue overflows at RP 3. Furthermore, some packets are lost due to collisions before they arrive at RP 2. This is because in the mushroom topology all PMs attempt to transmit to RP 3, increasing the contention level around RP 3, giving RP 3 fewer opportunities to transmit to RP 2. Because we are interested in high quality ($> 90\%$ delivery ratio), this first result suggests that the upper bound on the number of PMs that MEDiSN can support, for this specific topology and PM transmission rate, is ten.

Next, we evaluate the effects of external interference by adding link noise from the noise trace collected at the Stanford Meyer Library, simulating interference from 802.11 networks [Lee et al. 2007]. As Figure 5 suggests, when the RPs do not retransmit lost packets, performance deteriorates quickly as the number of PMs increases. On the other hand, retransmissions can mask the effects of interference to the point where the average reception rate does not differ appreciably from the perfect link case (cf. Figure 4). Finally, using 30 retransmissions, which is the default value of CTP, does not improve the average reception rate.

This lack of improvement can be explained by the results shown in the top panel of Figure 6. This graph presents the number of transmissions needed to successfully transmit a packet in perfect and noisy channels. It is apparent that most packets do not require 15 retransmissions. Thereby, having a maximum of 30 retransmissions will not provide any benefit. In fact, *the opposite* is true. As the bottom panel of the

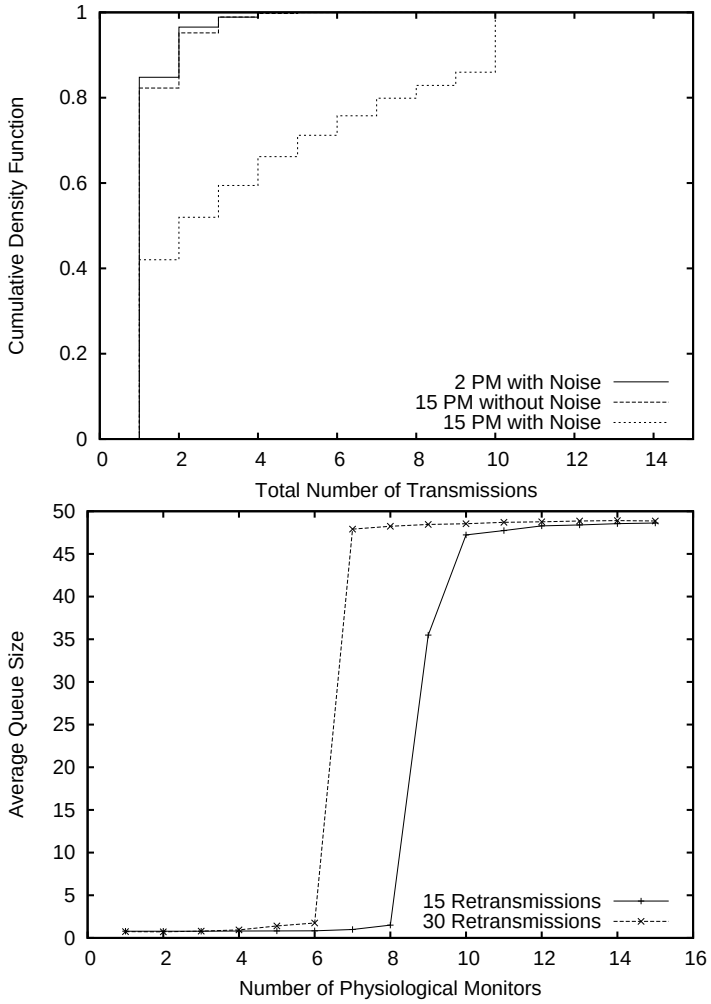

Fig. 6. (top) CDF of the number of transmissions for the mushroom topology with a maximum of 15 retransmissions. (bottom) Average queue size in RP 3 for the mushroom topology and noisy environment. The maximum queue size is 50.

same figure shows, as the number of PMs increases, the queue at RP 3 suddenly starts to overflow. This overflow happens earlier when the maximum number of retransmissions is set to 30. These overflows are due to packets that arrive while RP 3 attempts to transmit the packet at the head of its queue. In other words, attempting to send one packet, leads to multiple packets being dropped at the tail of the RP's queue.

Figure 6 also can be used as evidence that CTP, specifically its forwarding engine which controls the number of retransmissions and the inter-packet transmission intervals, is not well suited for high data rate networks. Next, we evaluate the benefits of the two mechanisms described in Section 3.2 on end-to-end performance.

Figure 7 presents the results with the proposed CTP modifications using the

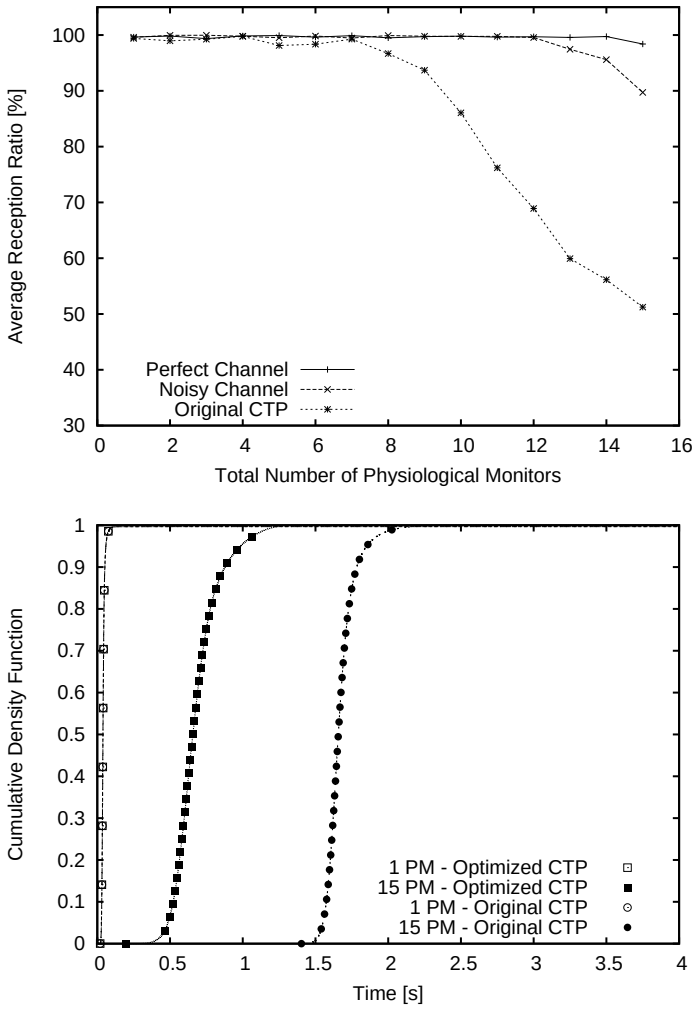

Fig. 7. (top) Average reception ratio (ARR) for optimized CTP in mushroom topology. (bottom) CDF of the end to end latency for the original and optimized CTP. The optimized CTP shows higher ARR along with lower latency values.

same conditions as those used in Figures 4 and 5. By controlling the number of retransmissions and dynamically adjusting the IPI, we reduce the number of packets dropped from the RP's queue and therefore achieve the performance shown in the top of Figure 7. Note that the maximum number of PMs supported, while maintaining delivery ratio $> 90\%$, increases by $\sim 50\%$ compared to that supported by the original CTP. Moreover, the delivery ratio achieved when a maximum of 15 PMs are active increases by $\sim 70\%$. The bottom graph of Figure 7 compares the end-to-end latency of the optimized and the original CTP. While comparable when the network is lightly loaded, end-to-end latency is reduced by threefold when using the optimized version of CTP in more loaded network settings.

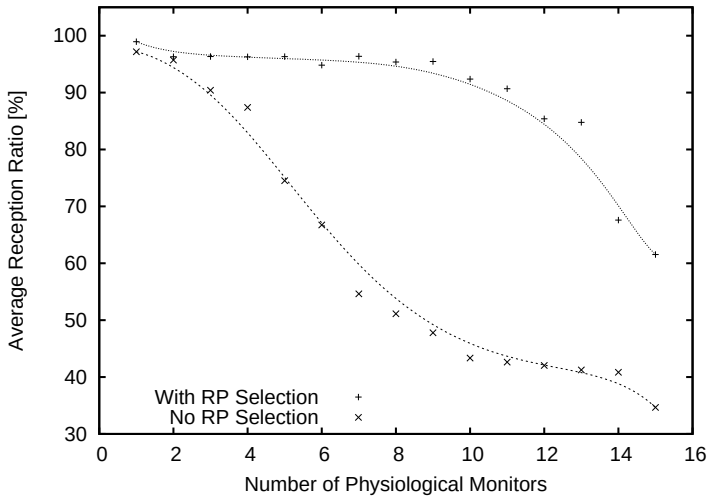

Fig. 8. Average reception ratio with and without RP selection for the line topology.

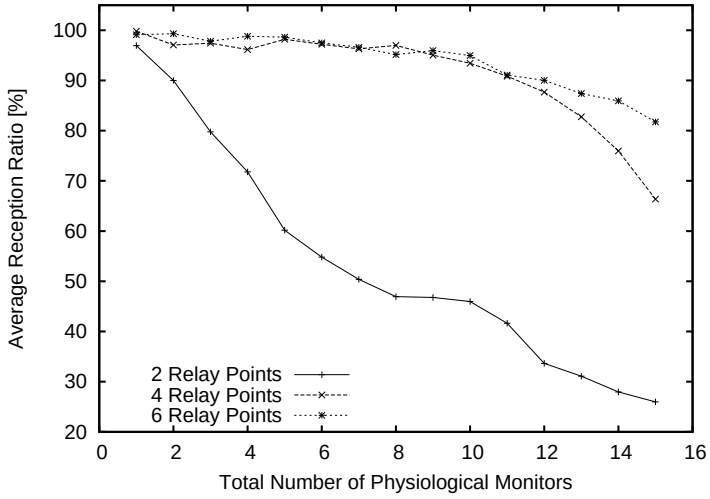

Fig. 9. Average reception ratio for line topology when an increasing number of RPs are distributed over the same distance.

4.1.3 *Line Topology.* We use the line topology shown in Figure 3, to compare the reception ratio with and without RP selection. As Figure 8 shows, RP selection provides considerable improvement by reducing the number of duplicate messages the RPs forward.

One of the benefits of having a separate RP infrastructure, is that this backbone can be properly engineered to offer high packet delivery ratios. In this respect, we evaluate the effect of RP placement on end-to-end performance. Specifically, Figure 9 presents the reception ratio for a line topology in which we place RPs increasingly closer to each other. The distance from the gateway to the farthest RP is fixed to 30 meters in all three cases. We add new RPs by decreasing the distance between neighboring RPs and use the log distance path loss model, with a

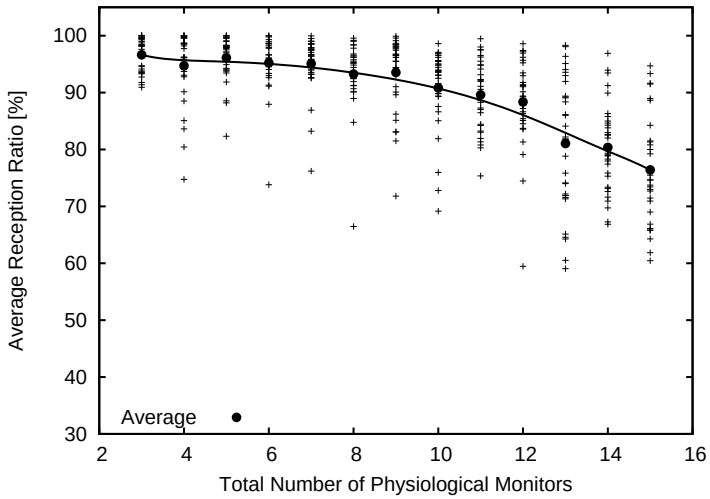

Fig. 10. Average reception ratio across random topologies with variable number of RPs. Different points represent the average reception ratio for a single experiment.

path loss exponent of four to generate the link attenuation levels [Rappaport 1996]. One can see that when the number of RPs is too small, the performance of the network degrades severely as the number of PMs increases. However, this problem can be addressed by adding more RPs. After a certain point however, adding more RPs does not improve performance as some are not used to route packets.

4.1.4 *Mesh Topology.* While the restricted topologies we used so far have helped us understand MEDiSN's behavior under different conditions, MEDiSN networks in reality will have more general topologies. To understand how MEDiSN will perform in such topologies we generate multiple random topologies in which 2 to 8 RPs and 3 to 15 PMs are randomly placed on a plane in a way that ensures that the network is connected. Figure 10 shows the results of this experiment. Each small point in the graph shows the average reception ratio for one test, while the larger points correspond to the average reception ratio over all experiments with the same number of PMs. The smooth line represents the system's overall trend and can be used to predict how the system will behave when a certain number of PMs are used. As in Figure 9, increasing the number of RPs improves MEDiSN's performance, even under more general topologies.

4.1.5 *Large Networks.* As previously argued, we target MEDiSN to both hospital deployments and disaster events. The latter ones are different in the sense that the number of patients will likely be higher but only critical information will be needed from each patient (i.e., no need for high-frequency ECG data). As a result, each PM will generate less data, only sending packets that contain blood oxygen and pulse rate measurements. According to the ISO 9919:2005 [ISO 2005] standard, clinical monitoring requires pulse oximetry samples to be sent at least once every 30 seconds. Given this requirement, we investigate the maximum number of such lower data rate PMs that MEDiSN can support. To do so, we generate a mesh topology with 20 RPs and up to 500 PMs. Each PM transmits one pulse oximetry

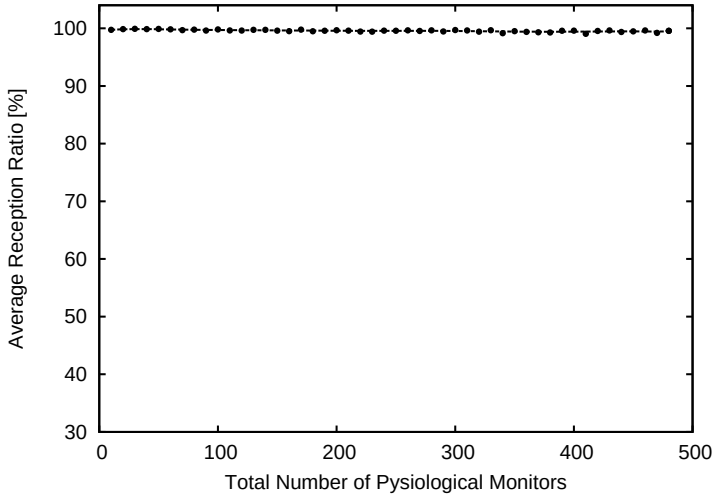

Fig. 11. Average reception ratio for a random topology with 20 RPs and up to 500 PMs. Each PM sends one pulse oximetry sample (5 bytes) every 30 seconds.

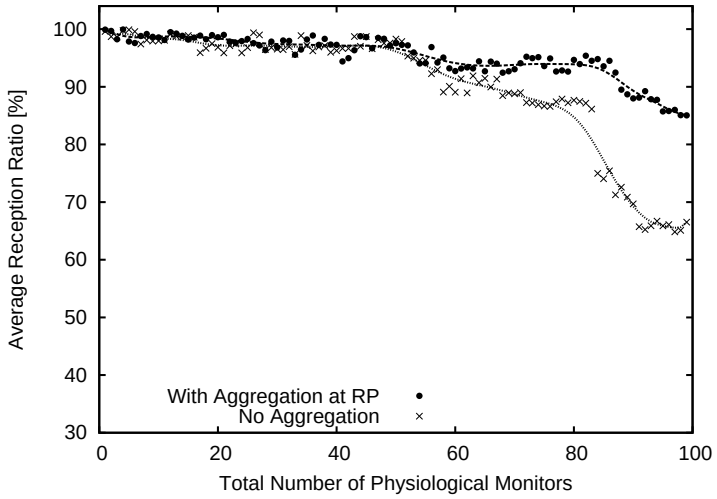

Fig. 12. Average reception ratio with and without aggregation. Each PM sends one pulse oximetry sample (5 bytes) per second.

and pulse rate sample, amounting to 5 bytes of application data, every 30 seconds. As Figure 11 shows, the average reception ratio stays above 99% for all scenarios. This result suggests that in addition to a small number of high data rate PMs, MEDiSN can support hundreds of lower data rate PMs, a condition that matches disaster response scenarios.

At the same time, trading off data rate to increase the number of supported PMs is, at best, a zero-sum game. Instead we are interested in techniques that increase the system's capacity, while keeping the offered traffic load constant. Message aggregation is one such technique. Specifically, RPs intercept PM packets and

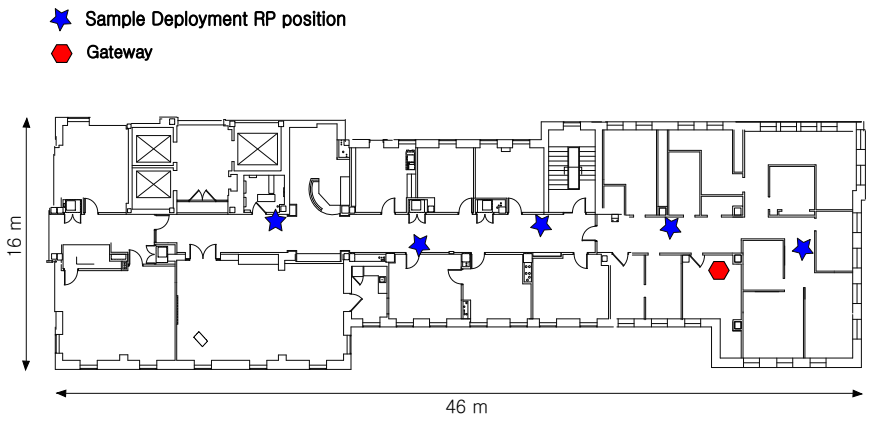

Fig. 13. Positions of relay points and gateway for the indoor testbed deployment.

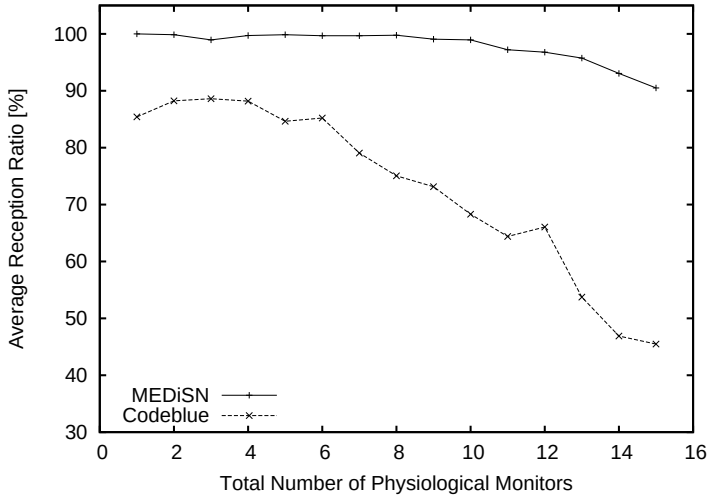

Fig. 14. Average reception ratio for MEDiSN and CodeBlue for a testbed deployment as a function of the number of PMs.

buffer them until a maximum-size packet can be sent or a given timer expires. Figure 12 shows the performance of this mechanism for a random topology with eight RPs and a variable number of PMs, each transmitting one pulse oximetry packet per second. Compared to the baseline behavior, aggregation is able to support 30% more PMs while maintaining $\geq$ 90% reception ratio. Furthermore, considering that the medical requirement is for one pulse oximetry sample per 30 seconds, the system can sustain even higher losses, or equivalently more PMs, while providing medically actionable data.

4.1.6 *Comparison with CodeBlue.* We evaluate the performance of MEDiSN in an indoors testbed equipped with Tmote Sky motes, which we use as RPs. The network includes an increasing number of stationary PMs connected to the

backbone of five RPs shown in Figure 13. We note that adding more RPs did not change the actual topology used to route the PMs' data.

We use the same network topology and mote hardware to compare MEDiSN with CodeBlue [Malan et al. 2004], using the latest software release publicly available from the official CodeBlue website. One might argue that MEDiSN has an unfair advantage over CodeBlue that uses PMs to also relay traffic. This however is not true. In fact, CodeBlue allows the use of separate repeater nodes that can relay the PMs' data. However, we found in practice that the performance of CodeBlue deteriorated when repeaters were employed. For this reason, the results we present for CodeBlue were derived by placing additional CodeBlue PMs in the locations where MEDiSN RPs were located. These PMs do not generate any traffic of their own, but can be potentially used to relay traffic. Moreover, unlike CodeBlue that defines multiple levels of network priority, the MEDiSN network treats all packets equally and deals with priorities at the application layer. For example, the PMs piggyback panic alerts to all of their outgoing messages while the panic button is pressed. Doing so, dramatically increases the probability of delivering these alerts despite faults such as lost packets and even RP failures.

We note that in this experiment we test the full MEDiSN system, with PMs encrypting and signing packets[2] before transmitting them to RPs. Figure 14 presents the results of this comparison. It is evident that MEDiSN outperforms CodeBlue both in terms of delivery ratio for the same number of PMs and in terms of the maximum number of supported PMs. Also note that the performance of MEDiSN in practice is similar to the one predicted by the previous simulation results (cf. Figure 7).

4.1.7 *Advantages of the dedicated wireless backbone.* MEDiSN differs from other medical sensing applications in its use of a dedicated wireless backbone. We devised an experiment to evaluate the quantitative benefits of this architectural decision. Specifically, we deployed a network of 30 RPs covering the first and the fourth floors of our office building. This setup mimics a wide-scale deployment over multiple hospital floors. For example, the Emergency Department at Johns Hopkins hospital includes an emergency room on the hospital's ground floor and an in-patient ward on the sixth floor. We use this network configuration to compare two alternatives. The proposed two-tier network infrastructure coupled with the RP selection scheme described in Section 3.2 and a "flat" network in which all nodes (i.e., RPs and PMs) participate in the CTP routing protocol. In both cases we record the packets delivered at the network's gateway from a mobile PM that travels at walking speed over a path spanning 1,000 feet across the building's two floors and the stairwell. At all times the PM was within the network's coverage area.

Figure 15 shows the packet reception distribution over time. Overall, the average reception ratio for the CTP-based PM and the MEDiSN PM were 73.99% and 96.43% respectively. The long sequences of consecutive packet losses in the top graph correspond to *outage events* during which the PM disconnects from its

[2]The inclusion of the CC2420 security header increased the length of packets exchanged during the testbed experiments by seven bytes. This additional header was not used in simulations as TOSSIM does not support security.

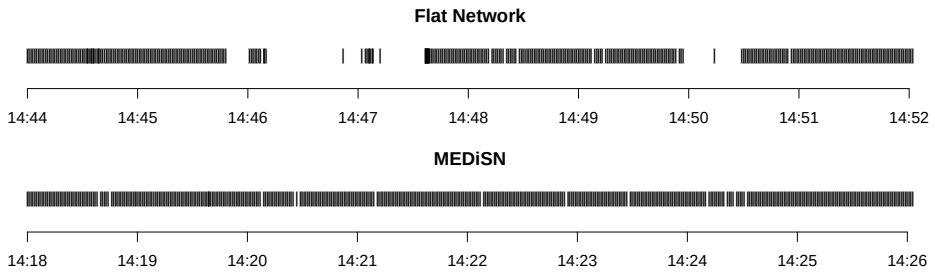

Fig. 15. Packet reception distribution during the multiple floor testbed experiment. Each short line corresponds to a packet reception with a flat network using CTP (top) and MEDiSN's two-tier routing (bottom) respectively. In both cases the PMs move ∼1,000 feet across the two floors. The two-tier routing mechanism coupled with the RP selection scheme improves delivery ratios for mobile PMs. The large sequences of consecutive losses for flat routing occur when the PM moves away from its previous parent node and attempts to reconnect to a new CTP parent.

	Average	Std. Dev.
Stationary Nodes	99.34	0.48
Mobile Nodes	96.37	1.08

Table I. Reception ratio statistics for stationary nodes and mobile nodes respectively during the multiple floor deployment. The side effects caused by mobility can be minimized using the simple RP selection strategy that PMs use to route their data to a backbone node.

previous CTP parent and attempts to find an alternate upstream neighbor. On the other hand, MEDiSN quickly reacts to RP disconnections through its robust RP selection mechanism. Finally, the average T_p measured at the mobile PM with CTP was 42.4 msec compared to 20.4 msec for MEDiSN. This difference is due to the fact that the PM which uses CTP attempts multiple retransmissions to its previous parent, while MEDiSN promptly switches to another RP once it detects loss of connectivity.

Finally, we compare the delivery ratios of mobile and static MEDiSN PMs. To do so, we randomly place three static PMs within the network's coverage area of 30 RPs while three other PMs were carried by volunteers walking randomly through the same coverage area for ∼30 minutes. Table I presents the average reception ratio for the two groups. While the reception ratio for the mobile group is lower, it is above our goal of 90% reception. Comparing these results to those reported for mobile PMs in CodeBlue [Shnayder et al. 2005] (50%-80%) suggests that the wireless backbone coupled with the proposed RP selection scheme allow MEDiSN to effectively mask the effects of mobility.

4.2 Hospital Deployment

We test MEDiSN in realistic environments through two pilot deployments in clinical settings and present lessons learned from these deployments.

4.2.1 *University of Maryland Shock Trauma Center.* We deployed a network of eight RPs covering the Operating Rooms (ORs) and the Post Anesthesia Care Unit (PACU) of the Shock Trauma Center at the University of Maryland Medical Center. The total area covered by our system, including all the ORs and the PACU, was approximately 10,000 square feet. We deployed the network through an iterative process during a walk-through of the Trauma Center. Specifically, we used the PM's LEDs to indicate when the PM moved out of the network's communication range and placed an additional RP where such disconnections occurred. LEDs on the RPs themselves were used to determine whether a RP successfully discovered a parent.

During three separate visits we installed up to eight concurrent PMs on patients who agreed to participate in our IRB-approved study. These PMs monitored the patients' vital signs (blood oxygen levels and pulse rate) from the time they entered the OR, throughout the operation, while they were moved from the OR to the PACU, and during their recovery period in the PACU. Depending on the patient, this process lasted from 90 minutes up to several hours.

MEDiSN succeeded in consistently monitoring the patients even when they were mobile. The average reception ratio over all patients was 98.25% with a standard deviation of 1.08. Moreover, we were able to control individual PMs remotely through MEDiSN's over-the-air management interface using the downstream messaging feature. We used this interface to turn off the PM's LCD and send commands to the patient when needed.

4.2.2 *Johns Hopkins Hospital Emergency Room.* MEDiSN was deployed for ten consecutive days, from 6 PM to midnight, within the waiting area of the emergency room (ER) at the Johns Hopkins hospital. A total of eight RPs were installed within the ER's waiting room, spanning an area of 6,500 square feet. Patients participating in our IRB-approved study were outfitted with a PM after their initial triage and the PMs were collected after they were admitted to the treatment area or when they left the ER. During the deployment, PMs with pulse oximetry sensors and LCDs were worn by 46 patients. On average, each PM transmitted 5,431 packets for each patient. An average of 244 packets were lost from each PM, resulting in an average ARR of 95.43% with a standard deviation of 5.41.

4.2.3 *Measurement Accuracy.* We validated the accuracy of the PMs' pulse oximetry data through a prospective observational study, comparing PM-derived measurements to a Nellcor OxiMax monitor, a monitor commonly used in clinical care. Five measurements were taken from each of 14 participants. The resulting data were used to calculate two correlations: one comparing the per-person average of our mote's oxygen saturation measurements to the average of Nellcor's oxygen saturation measurements and another for the pulse rate measurements. For oxygen saturation measurements, the data were correlated at $r(14) = 0.995, p < 0.01$, while for the pulse rate measurements they were correlated at $r(14) = 0.98, p < 0.01$.

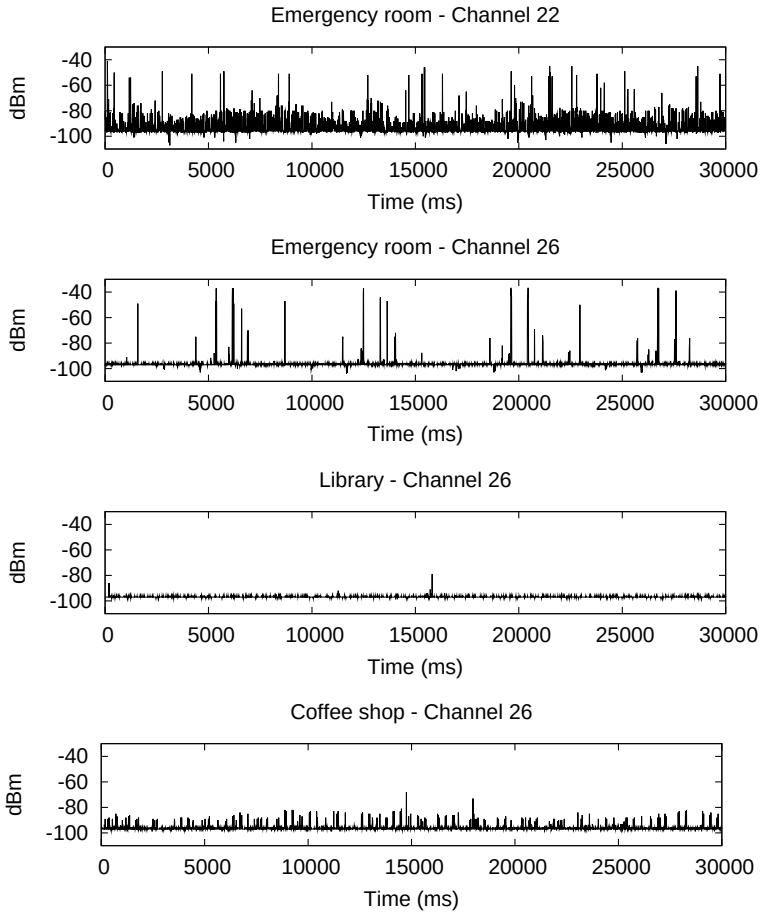

Fig. 16. 1 KHz channel noise samples collected from 802.15.4 channels 22 and 26 at the JHU Emergency Room, a university library, and a coffee shop for channel 26. Both ER channels exhibit pronounced noise spikes not seen in the other two environments.

4.2.4 *Experience from Hospital Environments.* We note that although the deployment areas were of moderate size, they both had multiple obstacles including glass walls, aluminum dividers, furniture, considerable human activity, thick steel doors, and lead painted walls. All these obstacles made the deployment of RPs more challenging than initially expected. However, the use of LED indications during the wireless backbone setup, as described above, helped ease the process. In practice, we found that the process of deploying the RP backbone at the Johns Hopkins ER required less than 30 minutes.

Another point to consider is the RF channel environment at the hospital. Like many other major hospitals both the University of Maryland Medical Center and the Johns Hopkins hospital have hospital-wide WiFi networks. Specifically, the WiFi network at the Johns Hopkins hospital used channels 1, 6, and 11. To avoid WiFi interference MEDiSN uses channel 26 from the 2.4 GHz frequency range of

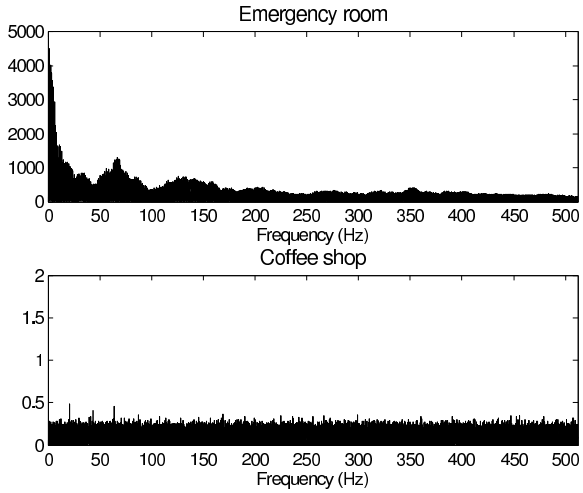

Fig. 17. Power spectra of the channel noise measurements from channel 26 at the ER (top) and an urban coffee shop (bottom). The ER waveform has multiple strong frequency components, significantly deviating from white noise seen on the bottom.

the IEEE 802.15.4 standard. Nonetheless, as the results from Figure 16 indicate, channel 26 at the ER had unexpectedly high levels of noise. Although the noise level in channel 26 is lower than the noise in channel 22 (that overlaps with WiFi), it is noticeably different from other channel 26 environments also shown in Figure 16.

In an attempt to understand the nature of this noise, we performed a Fourier transform of the traces from channel 26. The results of this analysis (shown in Figure 17) indicate that the trace obtained at the ER includes multiple periodic components, considerably deviating from white noise. Our current conjecture is that other wireless devices such as cordless phones are the root cause of this interference. Nonetheless, as the previously reported packet reception ratios prove, the combination of a well-provisioned backbone and occasional hop-by-hop retransmissions mask packet losses due to the environment and external interference.

4.3 Energy Consumption

We focus our attention on physiological monitors, as they are the only energy constrained devices in MEDiSN. We measure the energy consumption of two PM configurations: the PM depicted in Figure 2, equipped with a pulse oximetry sensor and an LCD display and another capable of performing ECG measurements but with no LCD display.

The ECG PM draws on average 20.4 mA at 3.3V when all of its components are active. Because in this case the radio is the biggest consumer of energy, a straightforward way to reduce the PM's energy consumption is to duty cycle its radio, using the approach described in Section 3. Doing so, results in a radio duty cycle of 6.6%, which in turn translates to an average current of 3.42 mA. This reduction can increase the PM's lifetime by almost sixfold. Considering a 1,200 mAh Li-Ion battery such as the one described in Section 3.1, the lifetime of the

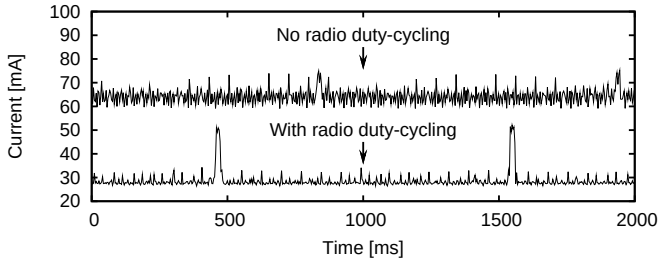

Fig. 18. Current draw traces for the pulse oximetry PM when the LCD display is turned off. The current spikes seen on the bottom graph indicate the periods of time when the radio turns on to transmit a packet. amounting to a 1.05% duty cycle.

ECG PM exceeds 14 days.

The PM with the pulse oximeter and the LCD (Figure 2) draws $\sim$ 110 mA at 3.7V, when all of its components are active. Turning the LCD off reduces the current draw to an average of 64.16 mA. Duty cycling the radio further reduces the average current draw. However, because the amount of data the PM sends is smaller in this case (one packet per second, because the pulse oximeter generates one sample per second), the resulting duty cycle is only 1.05%, reducing the average current draw to 28.46 mA (see Figure 18).

We measured experimentally the lifetime of the pulse oximeter PM to be approximately 10.9 hours when the radio and the LCD are constantly active. Duty cycling the radio and turning the LCD off while generating one packet per second increases the PM's lifetime to $\sim$ 42.16 hours (i.e., approximately 1 day and 18 hours). The next improvement in lifetime comes from turning off the pulse oximeter when it is not used. We do not duty cycle the pulse oximeter when generating one packet each second because the pulse oximeter requires $\sim$ 5 seconds to warm up before it generates an initial sample. However, we can duty cycle the pulse oximeter while generating one sample every 30 seconds to meet the requirements of the ISO 9919:2005 standard [ISO 2005]. In this case the radio's duty cycle decreases as well because the PM generates one packet every 30 seconds. In summary, these effects combined reduce the average current draw to 10.8 mA and thereby increase the PM's lifetime to $\sim$ 4.62 days. The next iteration of the hardware will address the discrepancy between the current maximum lifetime and the application requirement for 5 days of continuous operation by using a slightly larger Li-Ion battery.

Finally, these lifetime results further justify the decision to assign the task of data forwarding to a dedicated RP infrastructure. Not doing so, would require the PMs to keep their radios on to relay their neighbors' data. Doing so would significantly reduce the PMs' lifetime.

5. RELATED WORK

Several wearable technologies exist to continually monitor patient's physiological data. The Smart Shirt from Sensatex [Sensatex, Inc. 2007] is a wearable health monitoring device that integrates a number of sensors onto the Wearable Motherboard from Georgia Tech [Georgia Institute of Technology 2006]. Several other

e-textile technologies have been introduced that embed sensors in garments [DeVaul et al. 2003; Marculescu and Marculescu 2002; Martin et al. 2003]. Furthermore, the Lifeguard project at Stanford University is a physiological monitoring system that includes physiological sensors connected to a wearable device with built-in accelerometers that communicates to a base station over a Bluetooth link [NASA Astrobionics 2004]. The HealthGear project uses a similar architecture, concentrating medical data to a user's cellular phone [Flores-Mangas and Oliver 2005]. The CustoMed project at UCLA provides a "plug-and-play" approach to medical sensing in which hardware and software combinations can be customized to the needs of individual patients [Jafari et al. 2005]. Finally, the SMART project at MIT uses WiFi-equipped PDAs to collect and transmit patients' vital signs [Curtis et al. 2008]. All these projects are limited to single-hop wireless networks and focus on monitoring individuals. On the other hand, MEDiSN is designed to carry data from multiple patients in hospitals and supports multi-hop communications through a wireless backbone of relay points.

Our work is most closely related to the pioneering work performed by the CodeBlue [Gao et al. 2007; Malan et al. 2004] and AlarmNet projects [Wood et al. 2008]. While CodeBlue supports many-to-many communications over an ad-hoc multicast routing infrastructure, MEDiSN optimizes many-to-one and one-to-one communications over a dedicated wireless backbone. AlarmNet provides an infrastructure for collecting data from static and wearable sensors to monitor the behavior and extract higher-level patterns of people in assisted-living and residential environments. It includes sophisticated query, privacy control, and data analysis mechanisms and focuses on collecting longitudinal data from individuals using multiple modalities. On the other hand, MEDiSN focuses on shorter time horizons (i.e., hours to days) and larger number of patients.

Ruzzelli et al. recently proposed a multi-hop routing and MAC protocol for medical sensor networks that uses synchronized sleep schedules to conserve energy [Ruzzelli et al. 2007]. Instead, we use a two-level architecture in which physiological monitors (PMs) do not forward data and therefore can independently turn off their radios after transmitting their own data. At the same time, relay points are persistently busy forwarding packets and therefore it is infeasible to duty cycle their radios. Furthermore, the fact that PMs can be mobile further complicates the task of maintaining a consistent sleep schedule. The Zigbee Alliance has defined the *cluster-tree* topology in which a group of Zigbee routers form a multihop tree to forward data from a set of Zigbee end devices directly connected to them [ZigBee Alliance 2006]. On the other hand, Zigbee cluster-trees use TDMA to coordinate node transmissions and impose constraints on the depth and the size of the tree [Cuomo et al. 2008], while our approach requires no synchronization and scales to very large networks.

6. SUMMARY

We present MEDiSN, a hierarchical wireless sensor network for monitoring patients' physiological data. MEDiSN comprises a set of Physiological Monitors (PMs) which collect, encrypt and sign patients' physiological data (e.g., pulse oximetry, ECG, etc.) before transmitting them to a network of Relay Points (RPs). These RPs

self organize into a routing tree which reliably delivers periodic data and alerts from the PMs to the network gateway as well as management commands from the gateway to individual PMs. The gateway forwards collected data to a back-end server which persistently stores them and disseminates them to authenticated GUI clients. The design of MEDiSN's GUI and architecture were based on multiple iterations of feedback from hospital personnel and first responders who will be the end users of the system.

We evaluate MEDiSN through a comprehensive set of simulations, experiments in two indoor testbeds, and pilot hospital deployments. MEDiSN can support tens to at least five hundred PMs, depending on the amount of data each PM generates. One can further increase the number of supported PMs by introducing aggregation at the RP level. Moreover, we show that the system can minimize congestive losses and reduce end-to-end delay by threefold by dynamically adjusting the maximum number of retransmissions the RPs attempt and computing the optimal inter-packet interval. Furthermore, properly engineering the RP backbone can improve the packet delivery ratio up to threefold as well. Finally, preliminary results from pilot deployments in a hospital are encouraging: PMs collect physiological data measurements that are as accurate as commercial patient monitors, the system can be quickly deployed in the hospital, and supports multiple mobile PMs with high delivery ratios.

As part of our future work, we will investigate mechanisms to further improve MEDiSN's scalability, including the use of higher throughput wireless network technologies (e.g., WiFi) for the RP backbone and decrease the energy consumption of the PMs.

Acknowledgments

This material is based upon work partially supported by the National Science Foundation under grant #0855191 ('miSense') and the Department of Homeland Security through a grant to the National Center for the study of Preparedness and Catastrophic Event Response (PACER). We would like to thank Dr. Richard Rothman and the staff members at the Johns Hopkins Hospital Emergency Room for their help with the deployment. Amey Chinchorkar and Michael VanDaniker lead the development of the back-end server described in Section 3.4.

REFERENCES

Abate, J. E. 1967. Linear and Adaptive Delta Modulation. *Proceedings of the IEEE 55,* 3 (Mar.).

American Association of Colleges of Nursing. 2004. Nursing Shortage Fact Sheet. Available at: `http://www.aacn.nche.edu/Media/Backgrounders/shortagefacts.htm`.

American Hospital Association. 2005. The State of America's Hospitals - Taking the Pulse. Available at: `http://www.ahapolicyforum.org/ahapolicyforum/reports/`.

CNN. 2006. Death after two-hour ER wait ruled homicide.

Coalition for American Trauma Care. 2006. Action Needed to Bolster Nation's Emergency Care System.

Cuomo, F., Luna, S. D., Todorova, P., and Suihko, T. 2008. Topology Formation in IEEE 802.15.4: Cluster-Tree Characterization. In *Proceedings of the Sixth Annual IEEE International Conference on Pervasive Computing and Communications.* 276–281.

Curtis, D., E.Pino, Bailey, J., Shih, E., MS, J. W., Vinterbo, S., Stair, T., Guttag, J., Greenes, R., and Ohno-Machado, L. 2008. SMART – An Integrated, Wireless System for

Monitoring Unattended Patients. *Journal of the Americal Medical Informatics Association (JAMIA) 15,* 1 (Jan.).

DeVaul, J. G. R., Sung, M., and Pentland, A. 2003. Mithril 2003: applications and architecture. In *Proceedings of the Seventh IEEE International Symposium on Wearable Computers*.

Flores-Mangas, F. and Oliver, N. 2005. Healthgear: A real-time wearable system for monitoring and analyzing physiological signals. Tech. Rep. MSR-TR-2005-182, Microsoft Research. May.

Fonseca, R., Gnawali, O., Jamieson, K., and Levis, P. 2007. Four-Bit Wireless Link Estimation. In *Proceedings of the sixth workshop on Hot Topics in Networks (HotNets)*.

Fulford-Jones, T., Wei, G.-Y., and Welsh, M. 2004. A Portable, Low-Power, Wireless Two-Lead EKG System. In *Proceedings of the IEEE EMBS Annual International Conference*.

Gao, T., Massey, T., Selavo, L., Crawford, D., rong Chen, B., Lorincz, K., Shnayder, V., Hauenstein, L., Dabiri, F., Jeng, J., Chanmugam, A., White, D., Sarrafzadeh, M., and Welsh, M. 2007. The Advanced Health and Disaster Aid NetworkL A Light-Weight Wireless Medical System for Triage. *IEEE Transactions on Biomedical Circuits and Systems 1,* 3 (Sept.).

Georgia Institute of Technology. 2006. The Aware Home. Available at: `http://www.cc.gatech.edu/fce/ahri/projects/index.html`.

Gnawali, O., Fonseca, R., Jamieson, K., Moss, D., and Levis, P. 2009. Collection Tree Protocol. In *Proceedings of the 7th ACM Conference on Embedded Sensor Systems (SenSys)*.

Hopkins interNetworking Research Group (HiNRG). 2008. CC2420 Security Support for TinyOS 2. Available at `http://hinrg.cs.jhu.edu/git/?p=jgko/tinyos-2.x.git`.

ISO 2005. ISO 9919:2005 Medical electrical equipment – Particular requirements for the basic safety and essential performance of pulse oximeter equipment for medical use.

Jafari, R., Encarnacao, A., Zahoory, A., Brisk, P., Noshadi, H., and Sarrafzadeh, M. 2005. Wireless Sensor Networks For Health Monitoring. In *The Second ACM/IEEE International Conference on Mobile and Ubiquitous Systems*.

Jafari, R., Noshadi, H., Ghiasi, S., and Sarrafzadeh, M. 2006. Adaptive Electrocardiogram Feature Extraction on Distributed Embedded Systems. *IEEE Transactions on Parallel and Distributed Systems 17,* 8 (Aug.).

Ko, J., Gao, T., and Terzis, A. 2009. Empirical Study of a Medical Sensor Application in an Urban Emergency Department. In *Proceedings of International Conference in Body Area Networks (BodyNets)*.

Lee, H., Cerpa, A., and Levis, P. 2007. Improving Wireless Simulation Through Noise Modeling. In *Proceedings of the Sixth International Conference on Information Processing in Wireless Sensor Networks (IPSN'07)*.

Lin, S., Zhang, J., Zhou, G., Gu, L., Stankovic, J. A., and He, T. 2006. ATPC: Adaptive Transmission Power Control for Wireless Sensor Networks. In *Proceedings of the 4^{th} ACM Sensys Conference*.

Malan, D., Fulford-Jones, T., Welsh, M., and Moulton, S. 2004. CodeBlue: An Ad Hoc Sensor Network Infrastructure for Emergency Medical Care. In *Proceedings of the MobiSys 2004 Workshop on Applications of Mobile Embedded Systems (WAMES 2004)*.

Marculescu, P. K. D. and Marculescu, R. 2002. Challenges and opportunities in electronic textiles modeling and optimization. In *Proceedings of the 39th ACM/IEEE Design Automation Conference*.

Martin, J. E. T., Jones, M., and Shenoy, R. 2003. Towards a design framework for wearable electronic textiles. In *Proceedings of the Seventh IEEE International Symposium on Wearable Computers*.

MoteIV Corporation. 2007. Tmote Mini. Available at: `http://blog.moteiv.com/archives/2007/05/introducing_tmo.php`.

MSNBC News. 2006. Tired of waiting for the doctor? You are not alone. Available at: `http://www.msnbc.msn.com/id/15487676/page/2/`.

NASA Astrobionics. 2004. Lifeguard Vital Signs Monitoring System. Available at: `http://lifeguard.stanford.edu./lifeguard_flyer.pdf`.

Nellcor Puritan Bennet Inc. 2006. OxiMax NELL-1: OEM Pulse Oximetry Module. Available at: http://www.nellcor.com/Serv/manuals.aspx?ID=291.

Nygaard, R. and Haugland, D. 1998. Compressing ECG signals by piecewise polynomial approximation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*.

Rappaport, T. S. 1996. *Wireless Communications: Principles & Practices*. Prentice Hall.

Ruzzelli, A., Jurdak, R., O'Hare, G., and Stok, P. V. D. 2007. Energy-Efficient Multi-hop Medical Sensor Networking. In *Proceedings of the first International Workshop on Systems and Networking Support for Healthcare and Assisted Living Environments (HealthNet)*.

Selavo, L., Zhou, G., and Stankovic, J. 2006. Seemote: In-situ visualization and logging device for wireless sensor networks. In *Proceedings of the International Conference on Broadband Communications*.

Sensatex, Inc. 2007. Smartshirt System. Available at: http://www.sensatex.com/smartshirt.html.

Shnayder, V., rong Chen, B., Lorincz, K., Fulford-Jones, T. R. F., and Welsh, M. 2005. Sensor Networks for Medical Care. *Techical Report TR-08-05*.

Stark, P. 2006. Press Release: Stark Opening Remarks at Emergency Care Hearing. Available at: http://www.house.gov/stark/news/109th/pressreleases/20060727_emergencycare.htm.

The Center for Disease Control. 2005. Press Release: Visits to U.S. Emergency Departments at All-Time High; Number of Departments Shrinking. Available at: http://www.cdc.gov/od/oc/media/pressrel/r050526.htm⟩.

Wood, A., Stankovic, J., Virone, G., Selavo, L., He, Z., Cao, Q., Doan, T., Wu, Y., Fang, L., and Stoleru, R. 2008. Context-Aware Wireless Sensor Networks for Assisted Living and Residential Monitoring. *IEEE Network*.

ZigBee Alliance. 2006. ZigBee Specification. Available at: http://www.zigbee.org.