A WIRELESS TELEMEDICINE SYSTEM WITH EXTENDED REPORTING RANGE AND PRIORITY MESSAGING

Todd W. Polk, *Member, IEEE*, William P. Walker and Dinesh K. Bhatia, *Senior Member, IEEE*
Erik Jonsson School of Electrical Engineering and Computer Science
University of Texas at Dallas
email: {twp017000, wpw021000, dinesh}@utdallas.edu

***Abstract*-Extensions and enhancements are proposed for a previously developed system to remotely monitor patient vital signs. This previously developed network utilizes Crossbow MICAz motes to create a wireless network to gather data, which is sent to a central monitoring station. It includes a graphical user interface to store and display incoming measurements for all patients being monitored. Here the range of the network is extended by interfacing a GSM modem to the wireless sensor base station, allowing critical data to be forwarded anywhere in the world and providing a remote query mechanism via the existing cellular infrastructure. Text messaging using Short Message Service is used as the communication interface. A priority message handling layer is added to the current protocol to insure delivery of critical data to the monitoring station and from there to medical personnel via the existing cellular infrastructure.**

I. INTRODUCTION

The advent of low-cost and low-power wireless sensor hardware is driving the development of applications in several industries, including the medical field. Of particular interest here is the ability to remotely monitor vital patient data [1]. Wireless sensor networks (WSNs) are emerging as the preferred choice for ad-hoc, large networks and appear to be a perfect fit for remote patient monitoring. The issue here is that wireless sensors have a very limited communication range (approximately 10m). Data is forwarded through a WSN to a central location, but to increase the range by which the data can travel requires an interface to a wider ranging network. WiFi and the Internet are two viable options; however both require the user to have access to the network. As widespread as they both are, neither can duplicate the "anywhere" coverage of the existing cellular infrastructure. For this reason, interfacing the WSN to the cellular network was chosen.

A successful telemedicine network based on wireless sensors has been developed and is described in [2]. Here we see a network based on Crossbow MICAz nodes that utilizes energy harvesting to self-power the routing nodes of a WSN [3]. The network allows the interface of multiple medical instruments to wirelessly collect patient data and forward that data to a central monitoring station. The monitoring station consists of a wireless node acting as the wireless network's base station and a host PC running a graphical user interface (GUI) to display the incoming data. We call this our "local network", as the range is limited to the immediate location of the wireless sensor nodes.

This paper describes our efforts to widen the range of our local network by developing a GSM interface to the existing system. The initial implementation allows the forwarding of critical readings from the local network to a care provider's cellular phone, and allows the care provider to remotely request data from the local network. All requests and information are passed using the established cellular Short Message Service (SMS) protocol. The system is further enhanced by adding a priority message layer to the existing routing protocol. This allows the system to insure successful delivery of critical messages from the patient to the central monitoring station.

II. SYSTEM DESCRIPTION

A conceptual view of the local network described in [2] is shown in Fig. 1. Each patient is connected to the wireless monitoring system, which allows the medical staff to track the patient's vital signs. The vital sign readings are transmitted wirelessly from the patient through a fixed infrastructure of routing nodes to a central monitoring system (also called the base station). Depending on the patient's distance from the base station, messages can pass

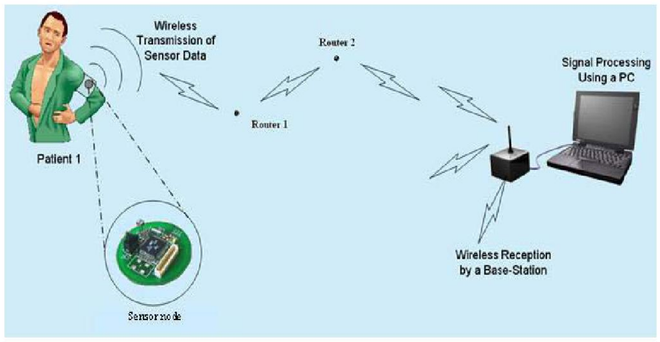

Wireless Transmission of Sensor Data
Router 2
Signal Processing Using a PC
Router 1
Patient 1
Wireless Reception by a Base-Station
Sensor node

Fig. 1. Local wireless monitoring system (Figure source: http://www.enel.ucalgary.ca/People/Haslett/WCLM/CCHE/WebPage/Vijay Devabhaktuni_Wireless_Proceedings.doc)

through multiple router nodes to reach the base station. The base station is connected to a host computer running a Java-based GUI to interpret, store and display the data.

The main limitation with this system is that the staff member needs to go to the monitoring station to view a patient's readings in the GUI. If a patient's vital signs are normal, this is not a big issue. However, if an abnormal reading is received, the system is confined to issuing a local warning that will only be seen by personnel at the monitoring station. In the case where an abnormal reading is possibly life threatening, a method to immediately inform the correct medical personnel is needed.

The range of the local network shown in Fig. 1 is extended by integrating a GSM modem to the system, and propagating the warning message out to the appropriate medical personnel so that they can respond to the emergency. Similarly, an authorized system user can remotely request information on a patient that is currently being monitored.

Fig. 2 shows the extension to the system. On the left side we see the wireless base station and host PC that comprise the monitoring station for the local network. Connected to this is a GSM modem, which can send and receive SMS text messages. The user can be anywhere – as long as cellular phone service is available.

This eliminates the requirement for medical personnel to stay in the immediate area to monitor critical patients. Alerts can be forwarded to them wherever they are. It also allows them to remotely query the system from their cell phone and receive updated information on any patient in the system.

A. <u>GSM Interface Hardware</u>

A commercially available Multi-Tech Systems MultiModem EDGE model GSM modem was connected via USB interface to the host PC [4]. It utilizes EDGE technology to deliver very high data rates, and is equipped with quad-band GSM to allow worldwide usage.

The modem was first accessed using the PC's HyperTerminal program to verify functionality and to send and receive several test SMS messages. Initial testing was successful, so our attention turned to integrating the modem into the existing GUI.

B. <u>GSM Interface Software</u>

The GUI used by the local network is Java based, and is described in detail in [2]. Here we needed to enhance the GUI to allow it to control the GSM modem as needed, and to monitor it to act upon any incoming messages. This was accomplished by modifying an existing Java library of SMS routines to our system's specifications.

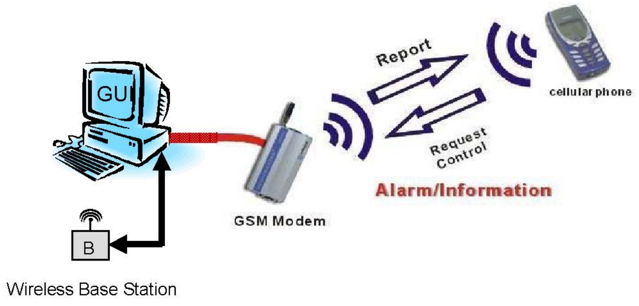


Fig. 2. GSM System Extension

SMSLib was modified to meet the needs of our system [5]. This library of SMS send and receive routines fit into our existing Java system with no problems. The required routines were updated with parameters specific to our modem and system. Two main threads were added to the GUI code – one to send messages when required and a second to monitor the modem for any incoming messages.

The send message thread is called when an incoming message from the wireless sensor network contains a data value that exceeds one of the preset limits in the GUI. In Fig. 3 we see an example of the patient information screen in the GUI. In the lower left hand corner of the window the limits currently set for the patient are displayed and can be adjusted. Any reading received that is outside these limits will trigger the sending of an alert message to the designated medical personnel.

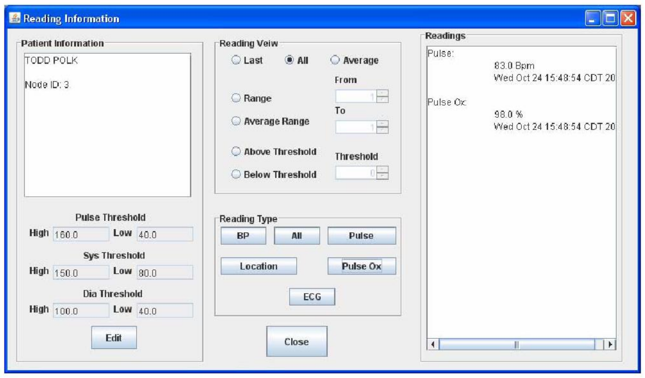
The send message thread is also called anytime an incoming data message from the wireless network is tagged as a priority message. Again, this will immediately cause the reading to be forwarded via the GSM modem.

The receive message thread runs continuously and polls the GSM modem to check for incoming messages. The current version of the system only allows a user to request the most recent data reading for a given patient. The user sends a text message to the system from their cell phone. The message contains the ID for the patient they are inquiring about.

The GUI will receive the request from the receive message thread, look up the last reading for the patient specified in the text message, and then call the send message thread to send the data back to the user. If the user requests information on a patient that is not in the system, a text message will be sent back to inform him of the error.

C. <u>Updates to Medical Interface Sensor Nodes</u>

While reading limits can be set in the GUI for any patient (as shown in Fig. 2), it is sometimes desirable to create an initial set of limits at the sensor node itself. In order to support this, changes were made to the interface programs for the different sensor nodes.



TODD POLK

Node ID: 3

Pulse Threshold
High 160.0 Low 40.0
Sys Threshold
High 150.0 Low 80.0
Dia Threshold
High 100.0 Low 40.0

Last [✓ All] Average
Range
Average Range
Above Threshold
Below Threshold
From 1
To 1
Threshold 0

Pulse:
83.0 Bpm
Wed Oct 24 15:48:54 CDT 20
Pulse Ox:
98.0 %
Wed Oct 24 15:48:54 CDT 20

Fig. 3. – Patient Information Screen

All software for the sensor nodes was written in nesC [6], an extension to the C language written specifically for use with TinyOS, a compact operating system developed at UC-Berkeley for use with wireless sensor motes [7]. All programs were written and compiled on a Windows PC running Cygwin, a Linux emulator [8].

Every wireless message sent through the local network includes a field containing the message type for that particular message. The initial thought was to add a new field to the message that would contain a priority tag. However, adding a new field would have increased the size of every data message, not just the priority messages. This would negatively impact the network's throughput rate and was judged to be undesirable. The decision was made to additional message types to the current field, which would have no effect on the size of the message. A priority message type for each sensor was added to the array of message types available.

The program for each sensor node was then enhanced to two areas. First, the ability to set limits at programming time was added. During programming, the user can set limits or ranges for all reading types performed by the individual sensor. The second change involved modifying the sensor to now compare the latest reading values against these pre-set limits, and then correctly determine whether it needs to send the message as a standard data message or as a priority message. Proper handling of priority messages was considered imperative and required changes in the routing protocol.

D. Priority Message Handling in the Router Network

Wireless networks are prone to dropping messages for a variety of reasons. Many of these reasons are controllable, but some are not. In dealing with potentially vital patient data, it became clear that a definitive method of insuring message delivery was necessary for priority messages.

All messages in the current system require an acknowledgement. After a data message is sent, the sending node will start an internal timer. The receiving node will send an acknowledgement upon reception of the data message. This will complete the handshake cycle, and the sending node will stop the timer. On the other hand, if the timer in the sending node completes and an acknowledgement has not been received, the sending node will resend the data message. The number of times this occurs is adjustable within the program, and currently is set to one retry. After the first retry, the message will be dropped from the sending node's message queue.

For routine data messages the decision was made to accept the fact that messages are occasionally dropped in a wireless network. If a patient's pulse oximeter reading is 97% and his pulse rate is 75, missing this reading is not going to cause a problem, and certainly is not life threatening. On the other hand, if his blood oxygen reading is 80% and his pulse rate is 50, this is potentially life threatening. Losing this message could be very serious. The sensor node (based on the limits set during programming), would tag this message as a priority message by changing the standard pulse oximeter message type to a priority pulse oximeter message type.

In order to insure end to end delivery for priority data messages, the routing network protocol needed to be changed. Whereas non-priority messages were resent one time, a priority message is resent many times. Each time the sensor node restarts the internal timer and waits for an acknowledgement. After a programmed number of retries (currently set to 50), the sending node will assume that its network connection has been severed, and will immediately broadcast a network connection request. This request is similar to the one issued at startup time when the node is first attempting to join the network. In doing this the node is looking for another path to the monitoring station. When the node receives a connection reply, it will then send the priority data message through the new path.

Acknowledgements are required at each hop through the network. Once a sending node receives an acknowledgement, it removes the message from its queue and moves on. The receiving node now becomes the sending node, and the process is repeated. This continues all the way through the network to the base station, which will end the process by sending an acknowledgement to the last sending node one hop away.

III. Experimental Results

Testing began with the priority message handling. A system like the one shown in Fig. 4 was built. Here we see two router nodes, R1 and R2, that are one hop from (i.e., in communication with) the base station B. A single sensor node S is added to the system. The sensor node takes a reading and then attempts to forward it to the base station through the routing network. It broadcasts a connection request to the network, which was responded to by both R1 and R2. In this case, R1 was chosen to route the data. The

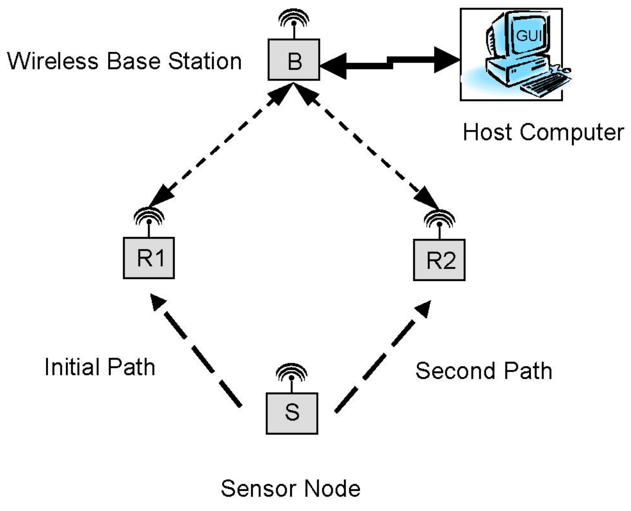


Fig. 4. Priority Message Handling

sensor node then forwarded the data message to R1 which in turn forwarded it to the base station for storage and display in the GUI.

The sensor node then attempted to send another normal data message through R1, which had been turned off. Since an acknowledgment was not received, S resent the message to R1 and then cleared it from the send queue. As expected, this message was not received and was therefore dropped.

Sensor node S then attempted to send a priority data message through R1. With R1 not responding, S continued to resend the message 50 times. After still not receiving an acknowledgement, S immediately broadcast a connection request. This was responded to by R2. S then sent the priority data message through R2 successfully and from there to the base station.

With the arrival of a priority data message, the GUI then proceeded to forward the message through the GSM modem to the author's cell phone. The message was successfully received. We were also able to request the most recent data reading from a patient by sending a text message with the patient's ID to the system from a cell phone. For instance, sending in the text message "3" would cause the system to access that patient's last reading and send the reply "Patient Todd Polk - Pox 96% Rate 75bpm". Sending a text message "7" (a nonexistent patient) would cause the system to send a reply "No patient with ID=7".

IV. Conclusions

A wireless telemedicine system is extended to allow critical alerts to be sent far beyond the boundaries of the standard system. The addition of a GSM modem to interface the system to the worldwide cellular infrastructure provides increased flexibility and capability. Medical personnel are immediately alerted to any serious condition wherever they are, and they also have the ability to query the system remotely.

Additionally, a priority message handler has been added to the base system. Through this the system insures the delivery of critical messages to the monitoring station. The GUI also forwards these critical messages through the GSM network to the care provider.

The current implementation, as discussed above, is the first version of the extended system. Further research will focus on integrating additional capabilities into the overall system. Our plans include:

- Allowing remote personnel to initiate readings in addition to the current query system.
- Developing a Smartphone interface to allow graphical viewing of patient data.
- The ability to adjust reading limits in the sensor from the monitoring station.

Acknowledgment

This research was supported by an internal grant from the University of Texas at Dallas (UTD) Erik Jonsson School of Engineering and Computer Science, and the work was conducted at UTD's Embedded and Adaptive Computing group (EACg) laboratory. The authors would like to thank UTD for the support.

References

[1] K. Lorincz, et al., "Sensor networks for emergency response: challenges and opportunities", *Pervasive Computing, IEEE*, vol.3, no.4, pp. 16-23, Oct.-Dec. 2004.
[2] A. Hande, T. Polk, W. Walker, and D. Bhatia, "Self-Powered Wireless Sensor Networks for Remote Patient Monitoring in Hospitals", *Sensors*, Vol. 6, Issue 9, pp. 1102-1117, September 2006.
[3] Crossbow Technology Inc., "MPR/MIB User's Manual", Rev. B, Document 7430-0021-06, 2005, http://www.xbow.com/Support/Support_pdf_files/MPR-MIB_Series_Users_Manual.pdf.
[4] Multi-Tech Systems, Inc., MultiModem EDGE Home Page, http://www.multitech.com/PRODUCTS/Families/MultiModem EDGE.
[5] SMSLib web site, http://smslib.org/.
[6] Gay, D.; Levis, P.; Culler, D.; Brewer, E., "nesC 1.1 Language Reference Manual," May 2003, http://www.tinyos.net/api/nesc/doc/ref.pdf.
[7] "TinyOS Getting Started Guide," Crossbow Technology, Inc., Rev. A, September 2005, http://www.xbow.com/Support/Support_pdf_files/Getting_Started_Guide.pdf.
[8] Cygwin Linux emulator for Microsoft Windows, http://cygwin.com/.