School of Engineering Science – Burnaby, BC – V5A 1S6 – proximus-ensc@sfu.ca

July 20th, 2006

Dr. Andrew Rawicz
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

Re: ENSC 440 Post-Mortem for a Wireless Monitoring System

Dear Dr. Rawicz,

The attached document details the entire process, including experiences of members within Proximus that were involved with creating the WMS (Wireless Monitoring System) within our ENSC 440 project course.

The current state of the device, pitfalls, as well as future developments are presented in this document. We discuss our deviations from the initial project proposal as well as the reasons for those deviations. Finally, the projected and actual timelines of creating the prototype of WMS are illustrated in this document.

Our company, Proximus, consists of 6 talented individuals: Zues Rawji, Kevin Ciarniello, David Liebich, Jatinder Singh Mann, Salman Abdollahi and Dallan Hunt, who have dedicated many hours into creating a viable solution in the medical industry which allows for better patient monitoring and more mobility to people when in need.

If you have any questions or concerns, please do not hesitate to contact us at proximus-ensc440@sfu.ca

Sincerely,

Zues Rawji
Chief Executive Officer

Enclosed: *Post Mortem for Wireless Patient Monitoring System (WMS)*

Post-Mortem for a Wireless Monitoring System

Project Team:

Zeus Rawji
Dave Liebich
Kevin Ciarniello
Salman Abdollahi
Jatinder Singh Mann
Dallan Hunt

Contact Person:

Zues Rawji
zrawji@sfu.ca

Submitted to:

Andrew Rawicz - ENSC 440
Steve Whitmore - ENSC 305
School of Engineering Science
Simon Fraser University

Issued Date: August 24th, 2006
Version 1.0

Table of Contents

List of Figures

List of Tables

Glossary

WMS	Wireless Monitoring System
DLM	Data Logging Module
GUI	Graphical User Interface
BLUI	Bluetooth User Interface
CRC	Cyclic Redundancy Check
ECG	Electrocardiograph
MFC	Microsoft Foundation Classes
PCB	Printed Circuit Board
ADC	Analog-To-Digital Converter

1 Introduction

For the past eight 8 months the team at Proximus has been researching and developing a *Wireless Monitoring System* (WMS). In this report, the trials and tribulations experienced throughout the design and development phases is discussed. In addition, any current issues and future design issues is also discussed in this report.

The WMS being developed will modularize the current system of sensor data acquisition from patients. By modularizing the current system, each individual component of the system can then be optimized and maintained increasing the efficiency and ease of monitoring patients by healthcare workers. In addition, by using wireless technology to monitor patients, patients are mobile and hospital care for these patients is extended to beyond the hospital bed.

After thorough research we feel that the WMS will offer the following benefits over existing solutions:

- Scalable
- Modularize sensor data acquisition
- Provide a medium for data mining for other applications
- Wireless connectivity
- Easy to use
- No line of sight communication required

Figure 1, as shown below, illustrates the system overview of the deliverable.

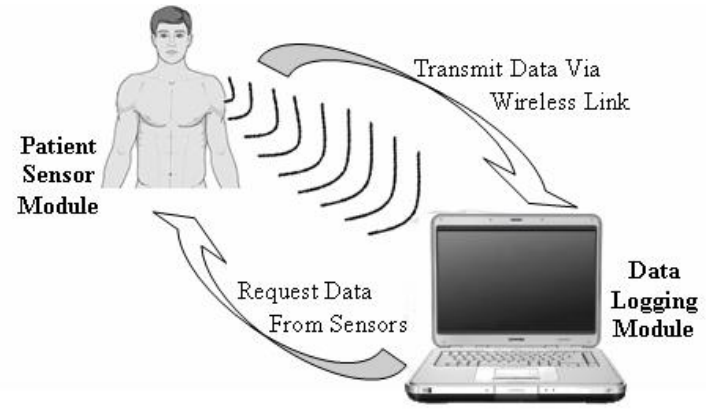

Figure 1 - Phase One System Overview (Data Logging Module taken from [1])

2 Current State of Device

2.1 General

The deliverable has the basic functionality that was laid out in the functional specification. Figure 2, as shown below, illustrates the system block diagram of the system delivered.

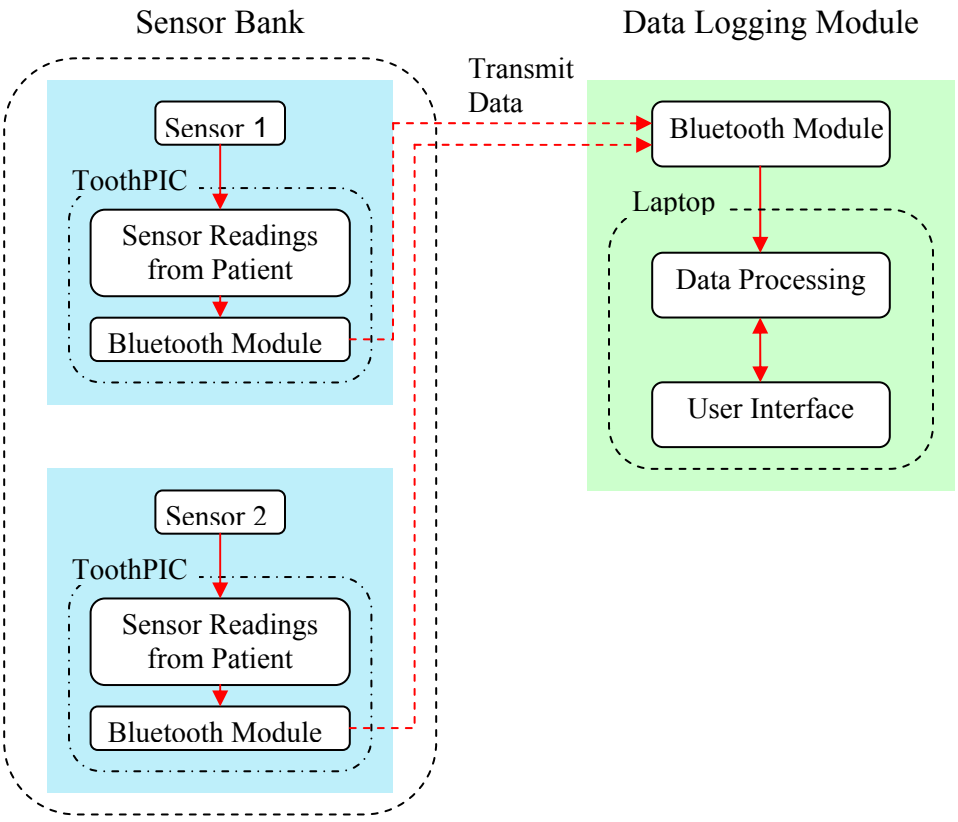

Figure 2: System Block Diagram for Phase One

The system consists of two modules: a sensor bank that is composed of multiple sensor modules worn by a patient, and the Data Logging Module (DLM) placed by the bedside.

The individual sensor modules within the sensor bank will transmit the data measured by the sensors to the DLM via a Bluetooth link. The DLM receives these measurements simultaneously from all sensor modules and performs the necessary error checking that is built into the Bluetooth protocol to ensure no data is corrupt. The data is then processed and placed into individual buffers on the DLM so that the Graphical User Interface (GUI) can read and display the data

accordingly. Medical staff will be given the choice of whether they wish to display this data in analog format (a wave), numerical format, or a combination of both.

The user interface works as follows for each module, either individually or in parallel:

- The user requests to open a connection to one of the sensor modules.
- After the connection has been established, the user sets the desired sampling parameters on the module via the GUI.
- Data transfer is initialized by the user and can be terminated at any desired time.
- The user requests to close the connection when they have acquired the desired sensor readings.

During the design process, input was taken from medical professionals, professors, and colleagues, in order to improve our design. Further development plans implementing these improvements will be discussed in section 4, Future Plans.

2.2 Software

The software portion of the deliverable is divided up into three categories: the Overall Software Design, Device-side Firmware, PC-side Firmware, and GUI. The GUI designed has been code named, the Bluetooth User Interface (BLUI). The deviances from the initial design for these three categories will be discussed in the subsequent sections, as well as the reasoning behind these changes.

2.2.1 Overall Software Design

The overall design for the software of the delivered system is shown below in Figure 3.

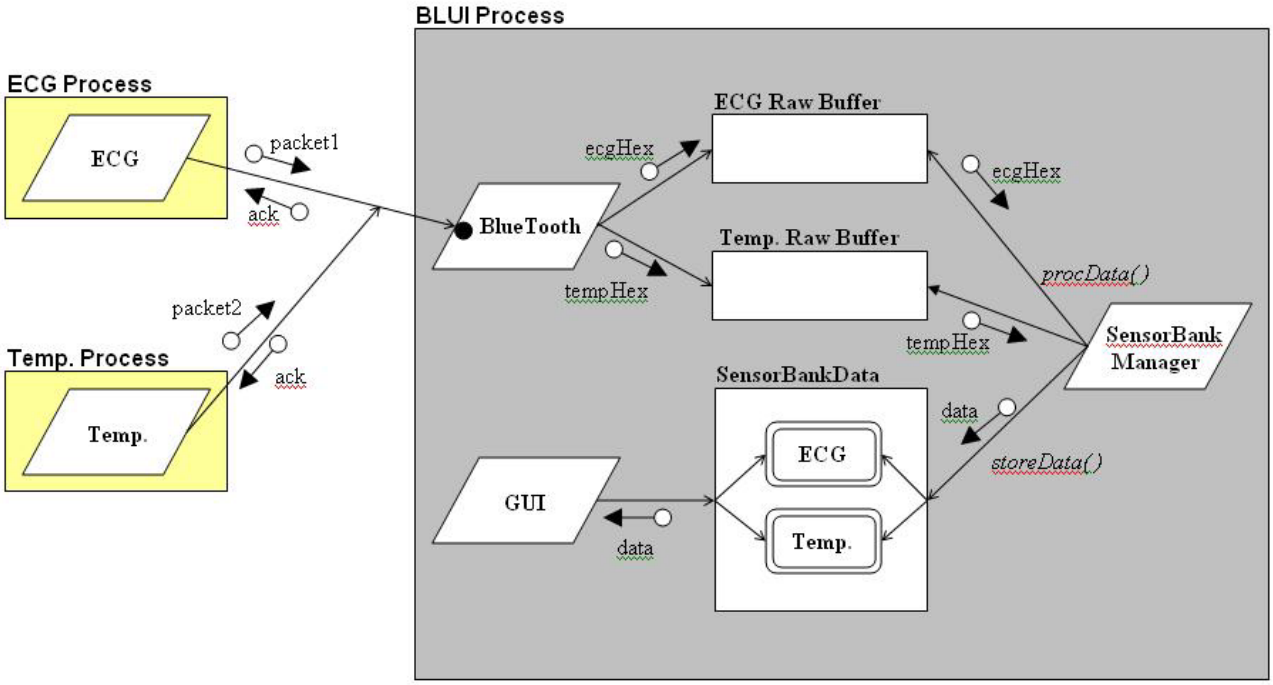

Figure 3 - Phase 1 Collaboration Diagram

A minor modification to the initial design was done in order to avoid the corruption of data when data was being stored in a main buffer. In the final design, two main buffers are created to store the data from two individual modules, as shown above in Figure 3. This design change marginally increases the static memory consumption of the software, however allows for a more scalable solution.

2.2.2 Device-side Firmware

The changes in the device-side firmware of the ToothPIC module can be attributed to a required firmware upgrade by Flexipanel Inc. in order to fix a critical software bug in the modules that were shipped to us. The new firmware upgrade allowed for a cleaner and much simpler algorithm to be used for data-acquisition. Below, in Figure 4, a flowchart illustrating the new algorithm is shown.

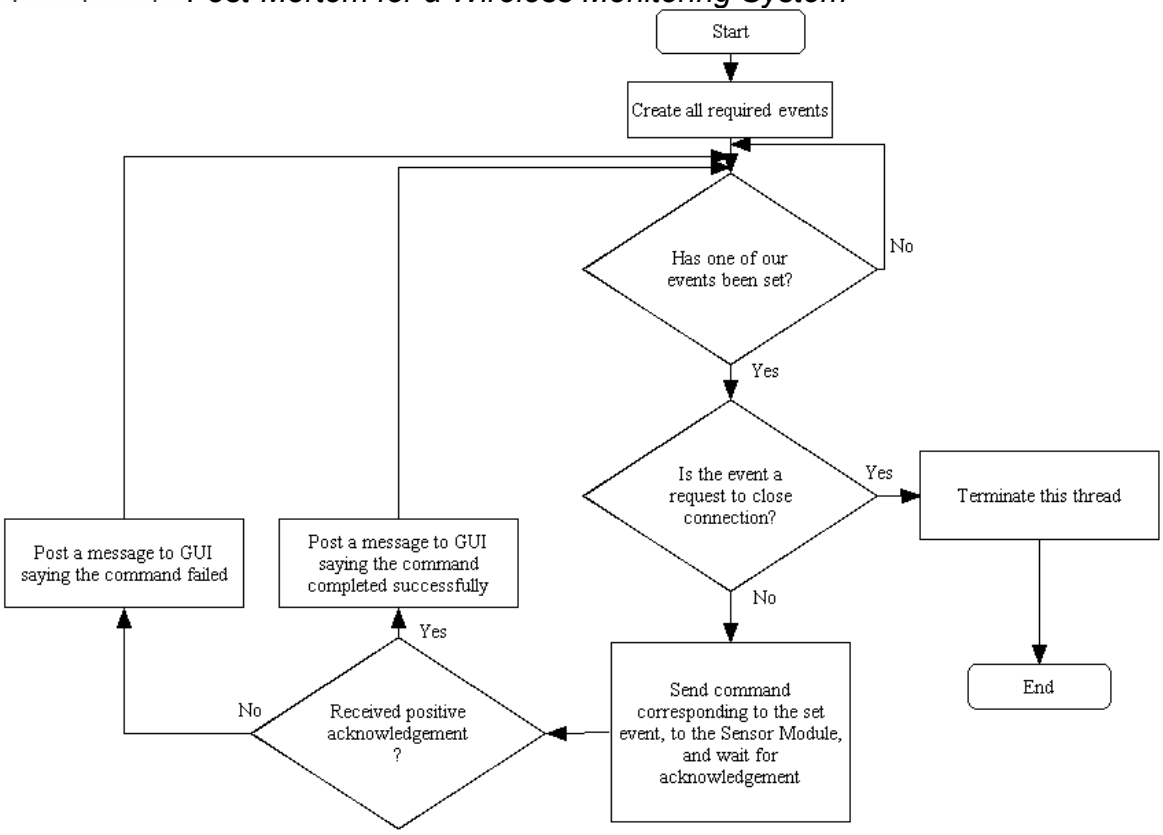

Figure 4 – Algorithm for Device-side Firmware

The ToothPIC acts as a slave which the master (DLM) controls. The DLM will issue commands to the ToothPIC module requesting to open/close a connection, initialize data logging parameters, and initialize/terminate data transfer.

The idea of further encapsulating the data packets into our own defined packet structure was removed since the Windows OS on the DLM can distinguish the source of the Bluetooth transmission it receives between various modules. Furthermore, the Bluetooth error checks were no longer required since Windows also verifies valid packet content before passing the data onto our application.

2.2.3 PC-side Firmware

2.2.3.1 General

The PC-side firmware was completely re-written to provide an easier software interface for the GUI to control the individual sensor modules through message passing. Four threads are created at this level in order to make the PC-side firmware robust and scaleable. These four threads are designed to allow parallel processing in order to increase efficiency of the system, and to allow for real time data transfer. This increase in efficiency results in a delay time well below what

functional requirement **R[66/C]** listed. In the subsequent sections, the four threads will be discussed.

2.2.3.2 Main Driver Thread

Below, in Figure 5, a flowchart illustrating the main driver thread is shown.

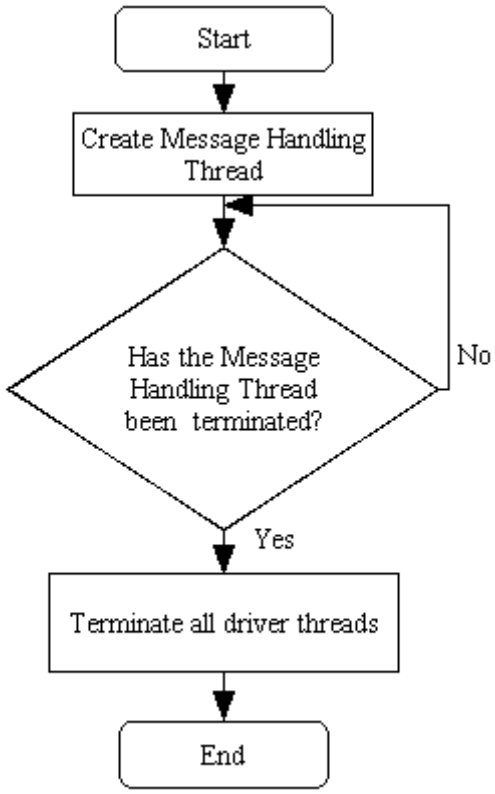

Figure 5 - Algorithm for the Main Driver Thread

Upon starting the BLUI application the main driver thread is created. This main driver thread is responsible for creating the message handling thread which will be discussed in the next section. Furthermore, the main driver thread is also responsible for terminating all threads, de-allocating stack space used by these threads, and closing all communication to the modules when the user wishes to exit the BLUI application.

2.2.3.3 Message Handling Thread

The entire message processing for commands sent by the GUI is done in the message handling thread. Furthermore, the message handling thread creates a serial thread for each module when a user requests to open a connection to each sensor. Below, in Figure 6, a flowchart illustrating the message handling thread algorithm is shown.

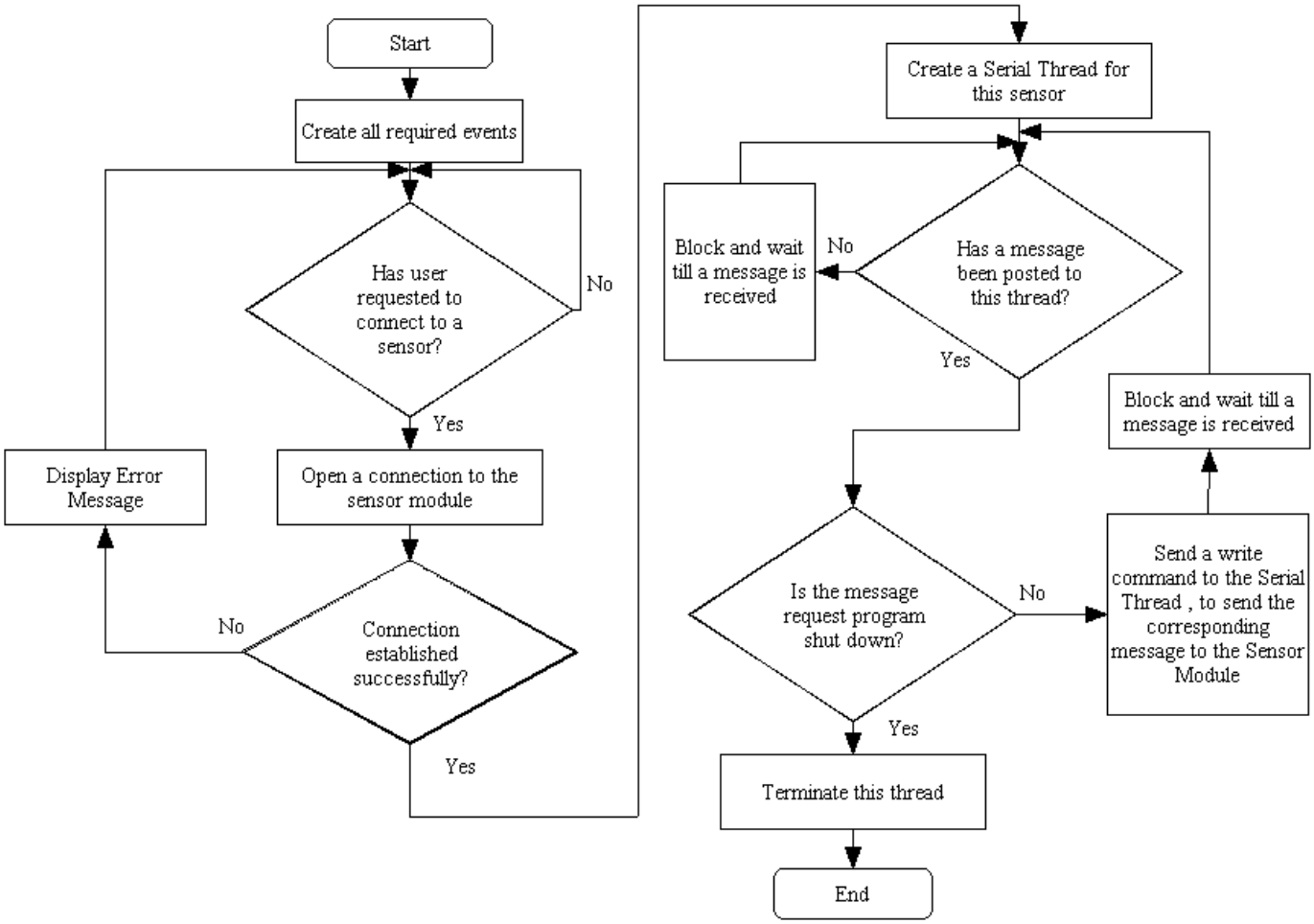

Figure 6 - Algorithm for Message Handling Thread

As additional sensors are added to the system the message handling thread will simply create a new serial thread for each additional sensor added. This design of the message handling thread greatly simplifies the overall system design as additional sensor modules can easily be inserted into the system with minor software modifications. The system is now scalable. The maximum number of sensor modules that can be connected at any time is seven, which is set by Windows.

2.2.3.4 Serial Threads

All serial threads created are identical to each other. Therefore, the algorithm discussed below applies to both serial threads. Below, in Figure 7, a flowchart illustrating the serial thread algorithm is shown.

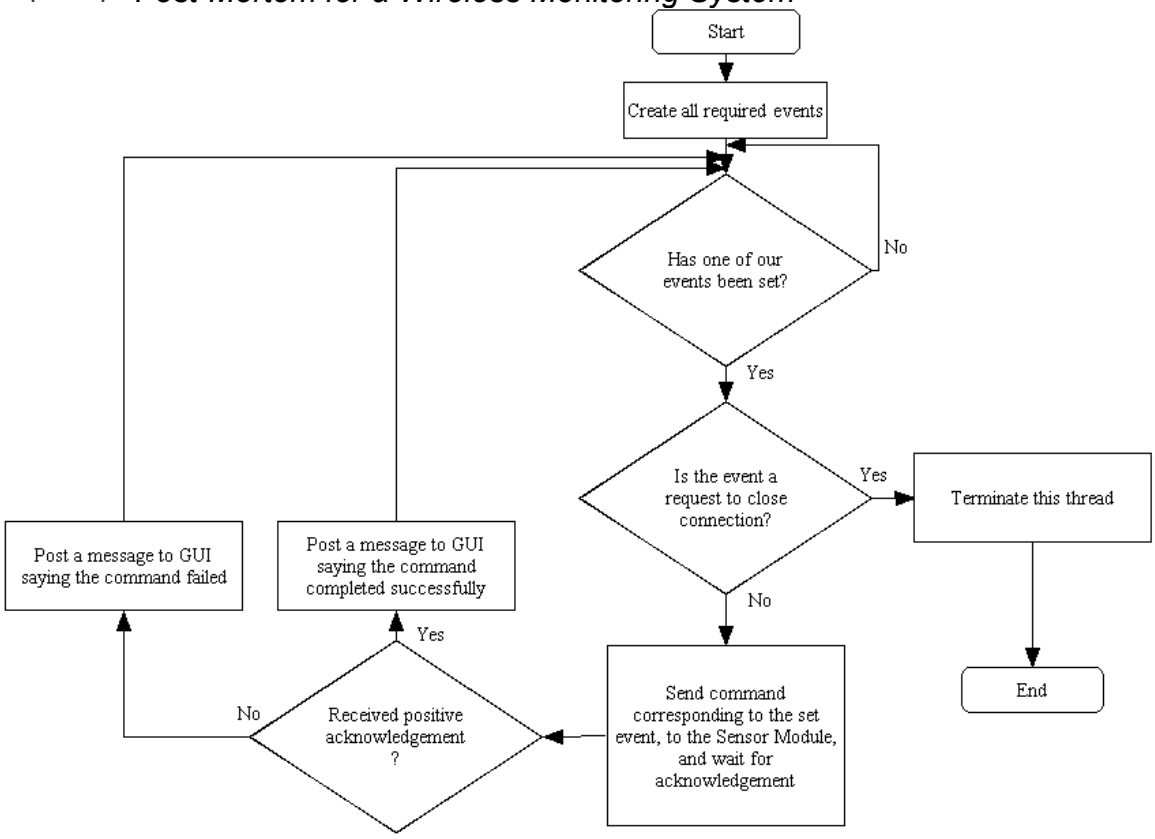

Figure 7 - Algorithm for the Serial Thread

The main task of the serial threads are to receive commands from the message handling thread, send them to the device, wait for an acknowledgement or data, and notify the GUI that the command was transmitted successfully. Also, when a user wishes to close a connection to a *specific* module, the serial threads have been designed so that the serial thread associated with the *specific* module can terminate itself.

2.2.3.5 Removal of the CRC Algorithm

Since the Windows OS used in our DLM verifies that the Bluetooth packet received contains valid data, the CRC algorithm illustrated in the design specification is not needed for phase 1 of the deliverable.

2.2.4 GUI

In this section the current state of the GUI will be discussed. In section 4, Future Plans, the future modifications that will implement input from medical professionals will also be discussed. These future modifications will make the GUI easier to use and reduce the amount of potential errors that could occur in the current design.

2.2.4.1 Appearance

The GUI currently displays sensor data from two different sensors. For Phase 1 we have confined the GUI to work for only 2 sensors to develop our proof-of-concept faster, rather than developing it for general display as would be the case for the final GUI.

For each sensor we display the following information:

- analog signal with compile time axis control
- digital signal
- module identification number
- connection status
- sample period (whether it has been set, as well as what it is in seconds)
- whether the analog input is set, and if it is being recorded from
- whether data is being streamed through the wireless connection, and
- whether the Red LED is on.

In addition to specific sensor information, we also include in the status bar the number of connected sensors compared to the total number of sensors.

Figure 8, as shown below, shows a screenshot of the current state of the GUI.

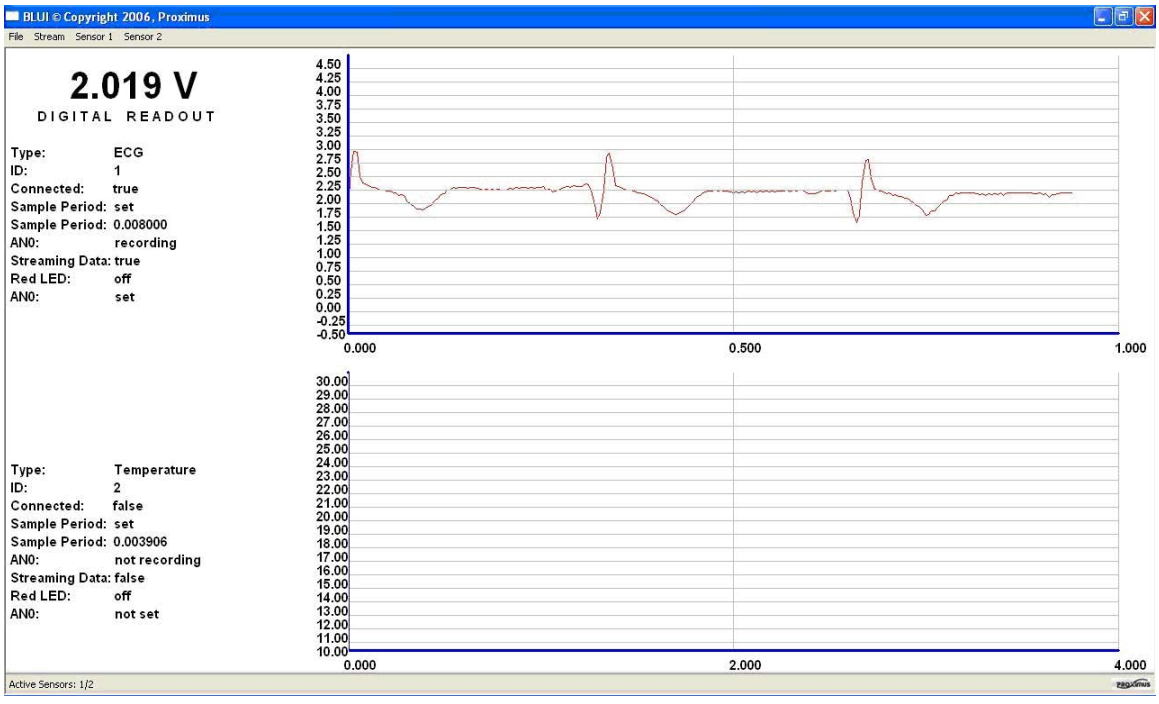

Figure 8 - GUI Screenshot

2.2.4.2 Deviations from Original Design

The original design requirements of the GUI are used, except for **R[77/A]** listed in the design specifications document. This requirement describes that the pulse rate of the individual should also be displayed. However, for a proof of concept of the WMS, we felt that additional processing of the data would not better illustrate the potential of the WMS. Furthermore, since the data is received as discrete voltage values, calculating the pulse rate would require significant mathematical calculations such as interpolating the voltage points and using Thompson's Algorithm. We decided that any further processing of raw data will be carried out in Phase 2 of the development where surveys would be conducted as to what type of information medical professionals would like to see on the final display.

Also, in our original design specification document we specify only a single GUI thread; we are actually using four threads now. The thread structure of the GUI is discussed in the next section.

2.2.4.3 Thread Architecture of the GUI

In our original design specification we mentioned only a single GUI thread. Through the design process we realized that one GUI thread would be insufficient, and so we actually use 3 additional threads (4 in total): The main GUI thread, a GUI worker, and two display threads. For each additional sensor module added to the system, an additional display thread is created. Figure 9, as shown below, illustrates the message flow between the threads.

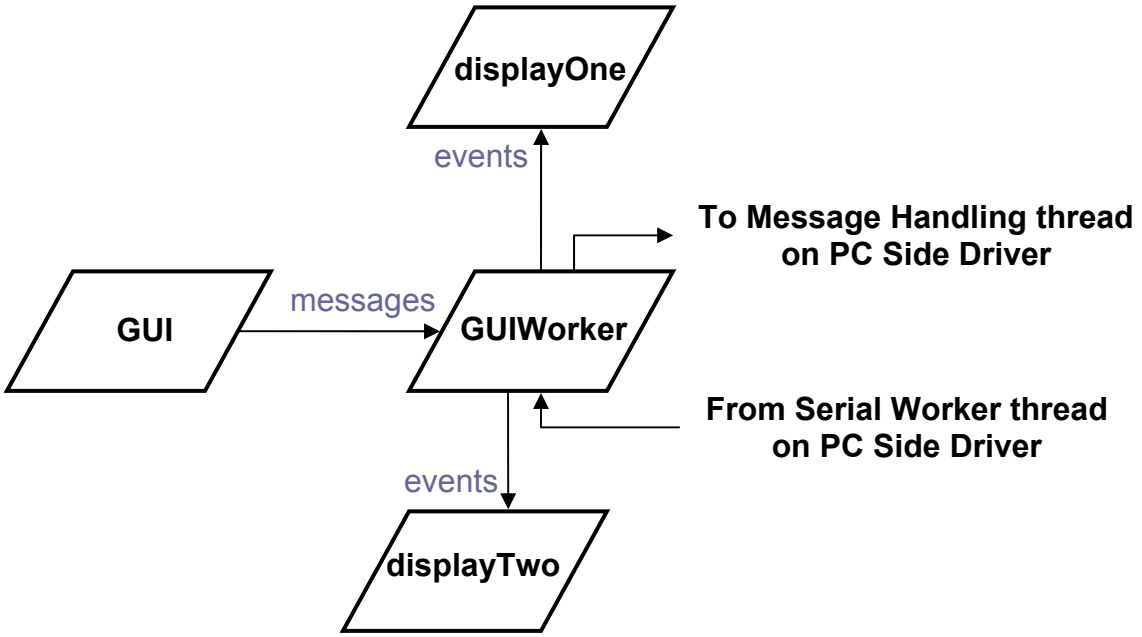

Figure 9 - Collaboration Diagram for GUI

The GUI thread is responsible for initializing the entire program: the window itself and all the data structures which the program uses. The GUI thread also is

notified of any user input and sends appropriate messages to the GUIWorker thread.

The GUIWorker thread processes any user input, thus allowing the GUI thread to be able to receive user input at any appropriate time. The main responsibility for the GUIWorker thread is to pass messages to the PC-side driver, and wait for acknowledgements from this driver indicating a completed task. The GUIWorker then updates the display accordingly.

Communicating display updates is done using displayOne and displayTwo. These threads are spawned separately whenever a connection is made to their respective sensor. Whenever data is received by the GUIWorker from the PC-side driver, an event is set to the correct display thread which fills data structures so that the GUI can display the most recent information upon the next refresh of the screen.

2.3 Hardware

In this section the current state of both sensors designed will be discussed.

2.3.1 Temperature Sensor

The circuit schematic for the temperature sensor is illustrated below in Figure 10.

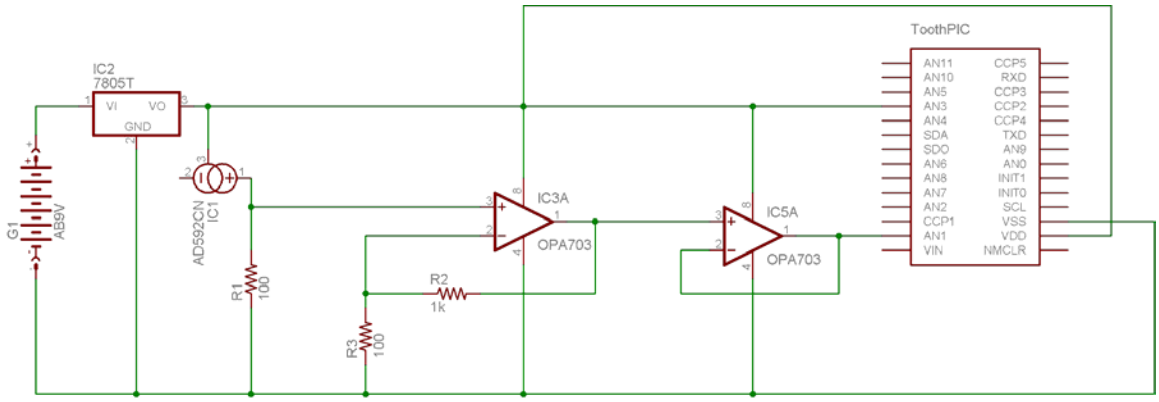

Figure 10 - Implemented Temperature Sensor for Phase 1 WMS

The temperature transducer is the AD592CN, while the op-amp used is an OPA703. The output current varies with temperature where we convert it to a voltage via resistor R1. The voltage is buffered to prevent loading of the measured variable. In addition, the buffer prevents a voltage that is too high from being applied to the input pin of the microcontroller.

The convenience of this transducer is its low cost and effective linearity. It boasts a 1 µA/°K and is perfect for measuring temperatures. Plus, it meets with specifications outlined in our functional specification design.

We added a gain buffer in order to make the voltage output readable into the ToothPIC. We have multiplied the gain in such a way that we are able to obtain a voltage level between -20 and 50°C accurately through linear relationships. This is done by amplifying it to 10mV/K. Therefore, if the current temperature is 25C, then the output is approximately 2.98V

In this configuration the temperature circuit meets the functional specifications requiring an accuracy of 2% (**R[25/A]**), and a sensitivity of 0.6 °C (**R[35/A]**). The following requirements are not met in the current state of the deliverable: **R[19/A]**, **R[22/A]**, **R[40/A]**.

Due to time and budget constraints we were not able to package the temperature circuit in a casing as it would be during mass production. We also did not add an acoustic warning signal because of time and budget constraints.

2.3.2 ECG Sensor

The electrocardiograph (ECG) circuit is a component of a Common-mode rejection circuit. The idea behind it is that it takes two signals and compares them, and if the two points are the same, it keeps that value and then removes all the other noise that is in the signal.

Three probes are placed on the patient. Figure 11, as shown below, illustrates the placement of the three probes that provide the most accurate signal from the human body.

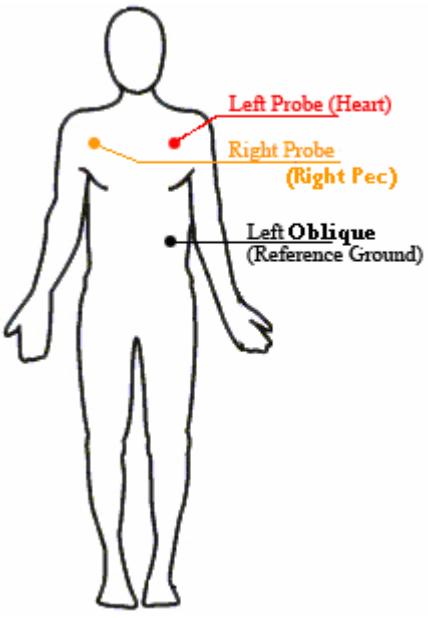

Figure 11 - Current Placement of ECG Probes

The left oblique acts as a reference point, where the other two probes are placed in order to have the signal pass through the heart. The final ECG circuit is shown in Figure 12 below.

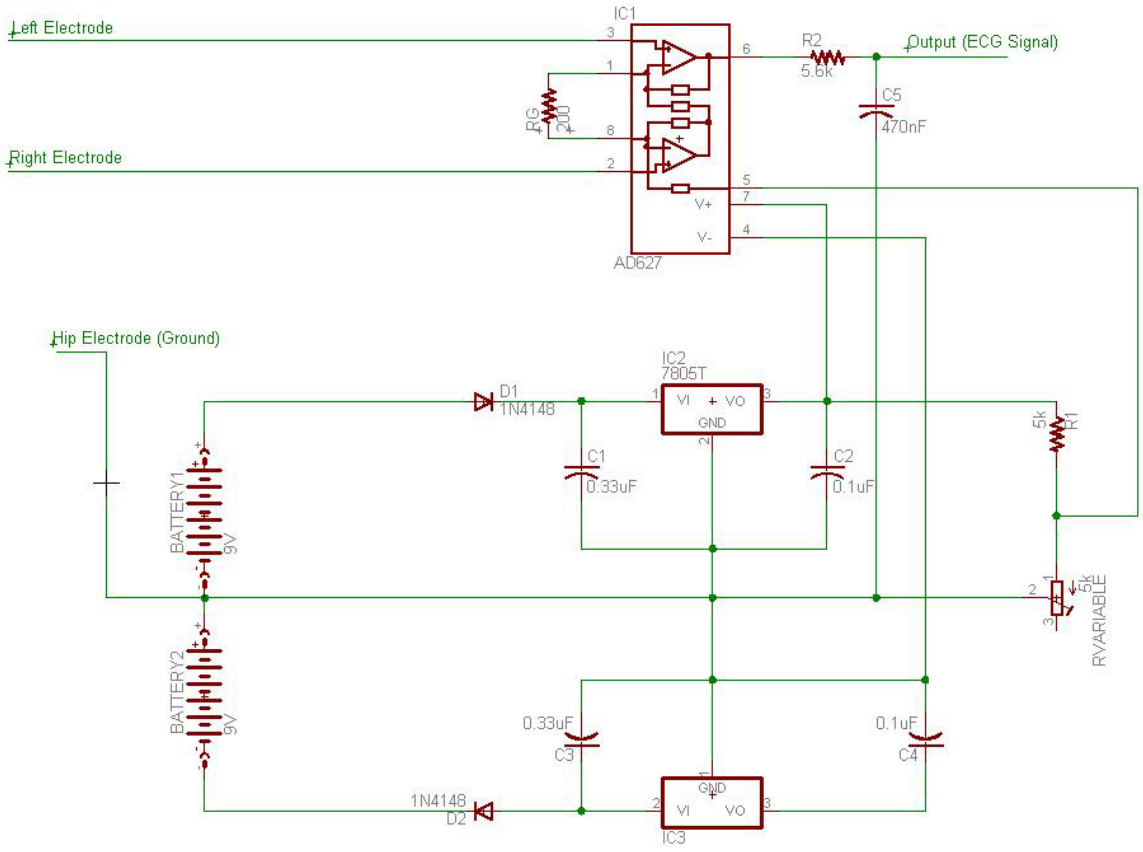

Figure 12 - Final ECG Circuit in Phase 1 WMS

Currently the batteries that are used are standard 9V batteries. Two energizer batteries that allow 900mAh are being used to power the circuit. With the addition of the Bluetooth module, we will need to obtain a more powerful battery that can power between 5 and 9V and give out 4500mAh; this would be for a full day that is consistently using the full range of the class 1 Bluetooth module.

In the end, the AD623 instrumentation amplifier was used because it has a programmable gain. The formula for the gain was

$$\text{Gain} = R_G / 200\ \text{k}\Omega + 5.$$

From the above equation, we were able to determine that if a 200Ω resistor was used, then a gain of 1000 would be obtained. The heart gives off a voltage around 2-4mV, therefore, the gain will make it so that the heart beat will be between 2-4V.

In order to power the AD623, two regulators and two batteries are needed in order to have -5V and +5V. Plus, another feature of the circuit is that a 20k trim pot is used in order to adjust the DC offset of the ECG signal, thus, it will be between the ranges of the input voltage for the ToothPIC module.

The ECG circuit only deviates from the following functional specifications: **R[38/C], R[39/C], R[40/A]**, **R[63/A], R[71/A], R[72/C]**. These requirements, due to time constraints, have been pushed to Phase 2 for the development of the WMS.

3 Remaining Issues

The deliverable works as presented in the previous section. However, there are still issues to be resolved. In this section, these issues will be discussed.

3.1 Device-side Firmware

After the ToothPIC module has completed streaming the desired sensor readings to the PC for the current session, the power for the module needs to be cycled before a new data streaming session can commence. In order to avoid cycling the power, the initialization parameters on the ToothPIC module need to be reset via software in order to restore the module to the initial startup state. The embedded software on the ToothPIC module will be modified in phase two of the project.

3.2 PC-side Firmware

When a command is sent to the ToothPIC module, the module responds with an acknowledgement indicating either success or failure . Since the form of communication is a serial stream of bytes, a chance exists that the PC-side Firmware may not receive the full stream of bytes indicating the status of completion from the ToothPIC module. Instead, the module may receive a part of the acknowledgement stream. In order to remedy this issue, a more complex handshaking algorithm needs to be implemented in order to verify acknowledgements sent to the PC. This algorithm will be implemented in phase 2.

3.3 GUI

There are 3 major remaining issues with the current GUI: invalidation, buffered display, and sample rate dialog box.

The first issue involves invalidation of the screen correctly; invalidation is when a portion of the screen is repainted. Currently, the GUI will sometimes loose its quality and color when one drags a dialog box across it and then closes it. In this case you may see the background disappear to white, and possibly loss of some analog signal display. The main reason for this problem is our inexperience of using Microsoft Foundation Classes (MFC) to develop a GUI. This GUI was one

of the first for our GUI design team and we learned a lot from doing it. One key issue we've learned is that the architectural design behind the GUI has to properly handle invalidation techniques to avoid loss of appearance.

The second issue involves lack of buffered display for each sensor. Buffered display would effectively solve the problem of analog signal display loss from incorrect invalidation. As it stands, for our analog display we keep track of only the last and current data values. The result is that all previous signal values that are cleared upon invalidation are lost. Buffering all the display points would enable us to redraw the entire analog signal every time the screen is invalidated.

The third issue involves a Microsoft bug relating to modal windows, namely the sample rate dialog box. Originally we planned on having it but we found out later that we wouldn't be able due to a Microsoft bug in Visual C++. It turns out that there is a bug, which causes problems irregularly, for a program which has two or more threads which display Modal dialog boxes. The chances of the bug surfacing increases with more threads and more dialog boxes.

The bug has been solved for .NET versions of Microsoft Visual C++ which we don't have, but none the less, we found a resolution technique[4] which unfortunately we could not use. If you spawn a thread to just handle modal dialogs then supposedly the problem will be fixed, but as you might guess, this would not work for concurrent modal dialogs because it would require two or more threads to display modal dialogs, which would return us back to our original problem. In our program we would have concurrent modal dialogs and so we would not be able to implement this resolution technique. The only option to solving this problem would be to acquire a more recent version of Microsoft Visual C++. This issue also prevented us from dynamically setting initialization parameters. Therefore, all parameters were set at compile time.

3.4 Hardware

Currently, one of the main problems with hardware is the use of a battery. Due to the cost and availability of a specific battery for the needs of this project, we were unable to find one that suited our specifications. Two 5V batteries that can allow 4500mAh would be ideal for this project.

Another problem that the hardware side faced was the PCB boards. The ToothPIC module that was bought from Flexipanel is wider than a standard 28 pin DIP. Therefore, finding a mount for the ToothPIC has been one of the challenges that WMS faces. The ToothPIC has to be directly soldered onto the PCB board. The problem with this case is that this is a $150 module and it would be more ideal in this phase to switch the module from circuit to circuit.

In order to provide casing for a module, realistic dimensions needed to be calculated. Since the sensor circuitry is currently going through PCB fabrication, we have decided to postpone casing until the PCB boards have been developed.

An additional problem that the hardware engineers of WMS faced was that they were unable to find a good wiring system between the ToothPIC and patient. In the current deliverable, unshielded wires with button clips are used. However, a proper system should be put in place. A solution to this problem would be to take coaxial cable and solder it onto a button clip and properly label them, since the left node has to be on the positive terminal, and the right node has to be on the right terminal. Either coaxial cable, or shielded wiring, can be used.

Further issues will be discussed in section 4, Future Plans.

4 Future Plans

Wireless Monitoring System can be greatly built upon. The foundation that has been built with the six individuals that are currently in Proximus has opened the envelope for more expansion. The developments that can be expanded on are described in the following sections. In addition, mistakes made in Phase 1 will be rectified in the next phase of development in order to provide a sound solution. These change in development plans are also discussed in the following sections.

4.1 Software

4.1.1 Platform Changes

After developing Phase 1 of the WMS, the software development team at Proximus has decided that in order to provide a more stable platform, QNX will be used as the OS for our future DLM. QNX is a real time operating system that provides a good basis for the processing of real time data, minimizing delay. The Windows OS used in Phase 1 was sufficient enough to provide a proof-of-concept, however the OS is not well equipped to handle real time data processing.

Furthermore, the software developed in Phase 1 of the WMS project will be ported over to run on the QNX OS. In addition, the software can also be ported over to a UNIX platform in order to run on the Central Location Module (CLM) which will be developed in Phase 2. Only the lower level driver will need to be re-written in order to handle TCP/IP packets, instead of a serial stream of data for the CLM.

4.1.2 Transmission Changes

In addition to the current error checking algorithms used by Bluetooth, the payload portion of the packet will be encrypted and a hash key will be passed along with each packet that will be used in order to decrypt the data on the DLM side. This will increase the security of data transfer, and provide a greater means to ensure patient privacy. This will especially be important for when Proximus decides to extend our WMS solution to at home patient care.

4.1.3 GUI Changes

Input was taken from medical professionals during Phase 1 of the project. This input will be implemented in Phase 2 to make the GUI more user-friendly. In addition, after further research, the GUI development team has decided that they will be switching over to a GUI developed in Java, as opposed to MFC.

A major modification would be to improve the appearance of our GUI by using appealing graphics and a better layout. Figure 13, as shown below, illustrates a possible Phase 2 GUI.

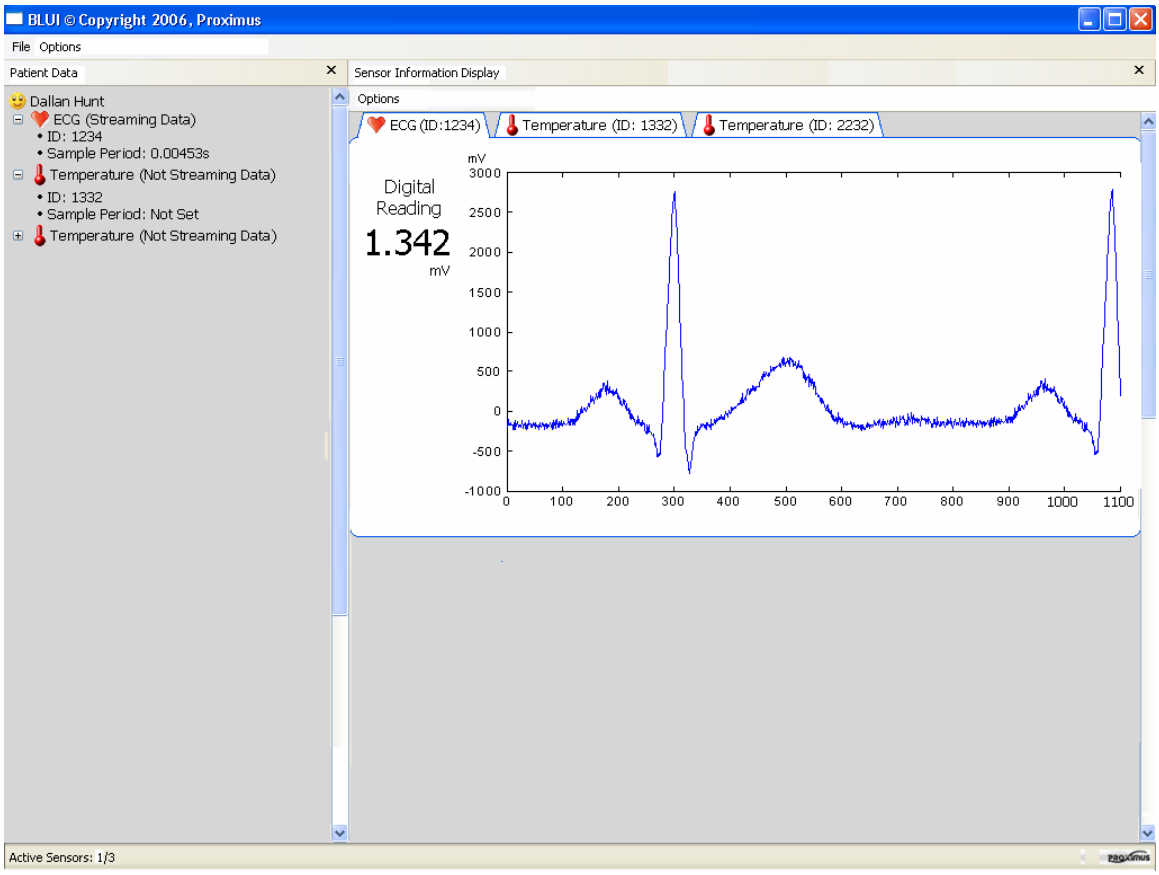

Figure 13 - Possible Improved BLUI

Notice the following:

- A tree list is used on the left side to organize a patient's information for each sensor; this will allow the user to expand and hide desired information for convenience.
- Sensor information is compacted to only the necessary information.
- Sensors are symbolized with small pictures for easy familiarity.
- The patient's sensors can alternatively be shown with tabs to hide unnecessary sensor information.

Another improvement would be to use Java. GUI development professional's state, working with Java is much more versatile for doing a GUI program. Implementing changes to the GUI is done much more easily when the GUI is developed in Java. GUI development time would decrease. Java is also platform independent which would make portability of our program easier.

This portability of the GUI will become important in Phase 2, when the GUI will be ported and modified in order to functional optimally for both the DLM and CLM. For the CLM, the code will be modified to accommodate for a network of patients whose data is received through a hospital's existing network (like Ethernet), and to run on UNIX. Much of the current code could be reused for the GUI of the CLM since the CLM would be a regular desktop PC. As for the DLM, the code would likely have to be ported over to run on QNX.

4.2 Hardware

Although a PCB for each sensor circuit is currently being fabricated, the hardware team still feels that additional hardware changes need to be implemented in order to provide an optimal solution. In addition, the hardware team has decided to shift their focus from sensor development to strictly development of the Bluetooth module in order to optimize the resolution of wireless transmission. The changes that need to be made in Phase 2 are listed below.

4.2.1 Bluetooth Module

- Research and develop a custom Bluetooth module.
- Reduce the current consumption of the Bluetooth module.
- Reduce the size of the module.
- Increase the resolution of the data transmitted.
- Incorporate a clock into the module in order to keep track of chronological order of data.

4.2.2 Additional Sensor Development

- Design a specification for 3rd party companies to develop sensors that can plug and play into our module.
- Create a physical interface for these 3rd party companies to plug their sensors into our module.

4.2.3 Sensor Casing

- Use lightweight casing that is durable and shock resistant to case the module and 3rd party developed sensor circuitry.
- Provide an interface for 3rd parties to insert their sensor circuitry into our case.
- Provide sockets in the casing for probes to plug into that are used by 3rd party sensor circuitry.

4.2.4 Batteries

- Provide an easy interface for faulty battery replacement
- Use rechargeable batteries
- Provide an interface to recharge batteries with.
- Use batteries that are light, small, and fit the functional requirements as listed in the functional specifications document.

4.2.5 Hardware User Interface

- Create a hardware interface (board) that users can plug the sensor module into in order to store in order to set connection parameters, patient name, and other desired parameters before initiating data transfer.
- The same user interface created should also be accessible by our troubleshooting staff through the internet, for rapid offsite troubleshooting.
- Strategically place LED's in order to indicate module status.
- Implement buttons in order to toggle the state of the module.

5 Budgetary and Time Constraints

5.1 Budget

The following table compares the estimated cost vs. the actual costs

.

Table 1 - Budget, Actual vs. Estimated

Module	Estimated Cost	Actual Cost
ECG Circuit	$25	$7.42
Temperature Circuit	N/A	$19.59
OP-AMPs and IC's	$75	N/A
Passive Components	$50	N/A
Bluetooth Module	$200	$435.71
Power Supply	N/A	$5
Bluetooth dongle	N/A	$66.92
BLUI	$0	$0
Development Firmware	$0	$0
Oxygen Rate Monitor	$225	N/A
Router	$40	N/A
Signal Booster	$50	N/A
LEDs	$10	N/A
Miscellaneous	$100	$65.38
Current Total	**$1125**	**$600.02**
%15 Overhead Estimate	$168.75	N/A
Total Budget	**$1293.75**	**$600.02**

The estimated and actual costs of WMS are $1293.75 and $600.02, respectively. The only cost that was underestimated was the Bluetooth module because it was initially believed that one Bluetooth module would be enough for our purpose. However, it was realized later that two Bluetooth modules would be needed. Plus, a burden of $95.30 in duties and taxes were placed on the shipping from the United States to Canada.

The reason that the total budget was overestimated was mainly because we didn't implement an oxygen rate monitor, router, or signal booster mentioned in our estimated cost.

Samples were obtained in order to bring down the cost for the prototyping of the product. To reduce the cost of the ECG circuit, components were used that were readily available, though not the best in size, they are good for the conceptual stage of WMS.

The temperature circuit was a cheap component, though the temperature transducer was not implemented onto a metal plate to increase surface area, though that wouldn't drive the cost up too much.

Though our estimated costs didn't involve Bluetooth dongles, it was quickly realized that they were needed for laptop support since the focus of the project was switched for Bluetooth on a laptop as opposed to a router, initially. Therefore, the cost of two Bluetooth dongles became $64.34.

The production costs associated for the final product will not include this cost. The production costs will only include the costs of the components associated with the Bluetooth module (ADC, Bluetooth antenna, PIC microcontroller), sensor pads, wiring, PCB layouts and components for ECG, temperature and other sensor circuits.

5.2 Time

The initial Gantt chart illustrated below, in Figure 14, shows the timeline which was projected by members of Proximus as a guideline to reach objectives and goals.

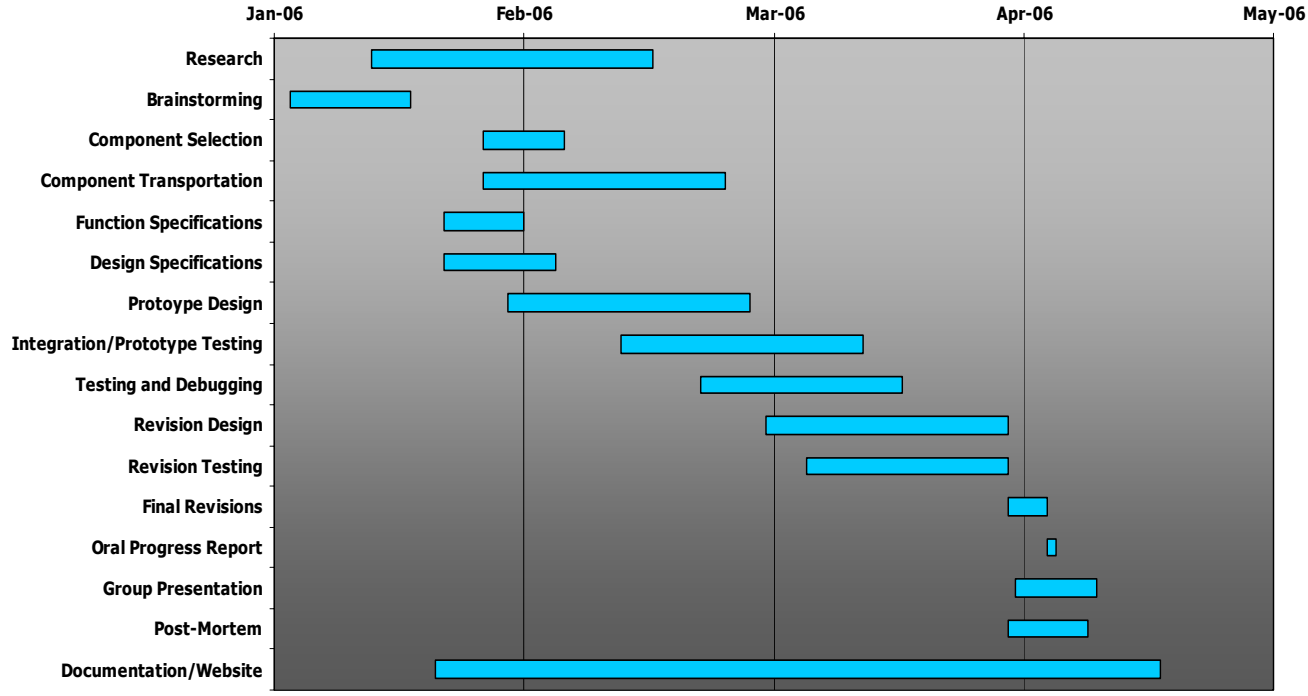

Figure 14 - Projected Schedule via Gantt Chart

Throughout the course of the project, sections were changed around and roles were changed. Thus, the focus of certain tasks was highlighted more than others. These tasks are shown in the actual schedule that has been displayed in the Gantt chart below. Here, component selection and transportation show a significant role, as design and building of circuits changed and so did the parts. Plus, obtaining a module which would serve the purpose of WMS took time to obtain.

The BLUI and the serial drivers could only be implemented as soon as the Bluetooth module was obtained. Therefore, those two parts were delayed more than expected. The ECG Circuit design took considerably longer than expected due to what was thought to be the complexity and the involvement to create. The ECG circuit run off batteries showed constraints to time issues.

Integration of BLUI and the serial drivers for the Bluetooth module required time in order to incorporate.

The actual Gantt chart is illustrated below in Figure 15.

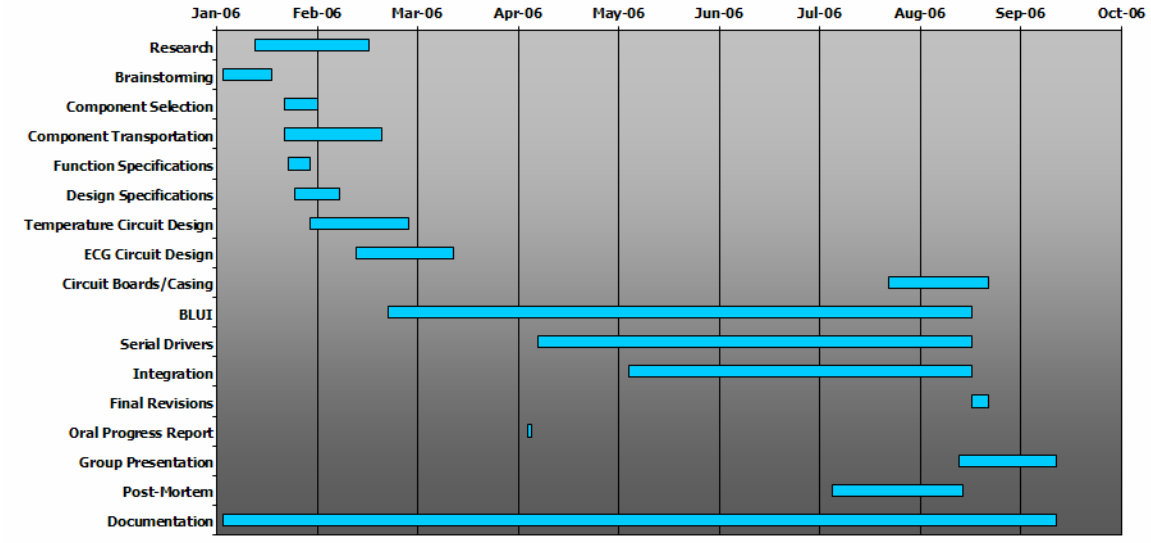

Figure 15 - Actual Schedule via Gantt Chart

6 Interpersonal and Technical Experiences

6.1 Zues Rawji

ENSC 440 gave me the opportunity to take on a leadership role in the design and development of an innovative wireless medical solution that other companies are starting to research and develop. As a result of seizing this leadership opportunity, I have gained invaluable experiences in team and resource management.

Being the most experienced person in the group and the team leader, I divided up the project into separate modules that could be designed and developed individually, and later integrated together with ease. Assigning these modules to various members was based on a blend of the desired skills they wished to learn, and their current technical expertise. I learned that a balance between both learning and accomplishments had to be achieved in order for all group members to be satisfied.

The most valuable experiences I gained from this project relates to group dynamics and leadership. When a member of your team is not performing at an expected level the leader must motivate the individual and address the issue instead of alienating the individual and shifting tasks around for the sole purpose of project completion. I also learned that when an individual just can't be motivated to complete their part, the leader has to create a back up plan in order get past this hurdle to reach the finish line. This usually involved in shifting around subtasks to other members of the team that took more initiative and were more motivated to complete the project.

On the technical side, I was the sole developer of the lower layer communication between the GUI and the ToothPIC module. Since I had previous experience in embedded development, I was able to apply these skills in a design that later proved to be robust, reliable, and easy to modify as additional features were added. This allowed for quick development once the GUI was fully functional. The technical experience I gained from this project mainly dealt with Windows driver development and the integration of an entire solution (both hardware and software). In addition, after assigning a task to a team member, I would check the end result and provide feedback on the deliverable.

All members of this team had the necessary technical expertise to complete this project. However, in my opinion not all members possessed the same initiative, dedication, and drive to achieve completion. For this reason, I've learned that forming a team with *driven* individuals who possess *some* technical skills in the required areas is more essential than forming a team with individuals who have the correct technical expertise, but are not as driven as others on the team.

In conclusion, I feel that ENSC 440 has allowed me to gain invaluable experiences in project management, team leadership, and some potential hurdles associated with a startup engineering firm. Since I plan on starting up my own firm in the near future, I feel these experiences will prove to be extremely valuable in avoiding costly mistakes.

6.2 Dallan T. Hunt

The 'ENSC 440' experience has taught me many things about working on a large group project. I learned about managing group dynamics, meeting deadlines, and handling an almost entirely self-directed project.

Throughout ENSC 440 there were conflicts with group members, changes in leadership roles, and poor communication. Although these negative aspects of the project happened, they did not happen all the time – of course there were positive aspects as well. But as I have learned, any negative quality can become a positive one if you learn from it, as I have done. I learned, among other things, dividing up work effectively among individual members or sub groups is only superseded in importance by choosing the correct work in the first place. I also learned that recognizing problems with members or sub-groups, and solving the problem as soon as possible is key to a successful group. Another thing I learned is that the group leader is extremely important. The leader must clearly communicate, be able to understand peoples' situations, and have a good sense of the strengths of members for effective workload distribution. Learning about group dynamics is extremely valuable as the knowledge can be used in further project work, but also in day to day living.

Deadlines, deadlines, deadlines! Previous to ENSC 440, I did have a clear understanding of deadlines from assignments and projects, but ENSC 440 added a flavor of realism. Although ENSC 440 was another university class, it seemed

that we were actually in a start-up company sometimes. The experience will be valuable since I have my own start-up company, and will likely work for a start-up company for the summer semester.

ENSC 440 is a self-directed project, and is unlike other university class since there is minimal professor assistance. As a result, I had to first decide what to learn, and then to learn it. This learning scheme was harder than it seems because I hadn't done something like it before on a project of this magnitude. I gained further knowledge on using MFC and basic multithreaded programming, as well as writing effective documentation.

I would like to thank everyone in my group for participating and creating all the experiences which happened, creating an excellent learning experience. Thank you!

6.3 David Liebich

As we were warned, our group was behind before we even started. In a group like ours, where most of us had never worked with the others before, it took us quite some time to adjust to each others working styles. Deadlines continued to be our most effective motivator as was witnessed by our productivity when various deadlines approached.

I have learned that in a group like ours that is not comprised of a large group of friends, it is beneficial to have a talented group leader. A truly talented group leader understands how to listen to the group member's concerns and then delegate the work in the most effective manner. The group leader also needs to understand that to encourage group member participation they need to use encouragement as well as discipline.

I also learned valuable lessons while designing the temperature sensing circuit. My first design was an optimal design that made maximum use of the analog to digital converter on the microcontroller. I simulated the circuit and it worked perfectly. It outperformed the minimum requirements from our functional specifications by a significant margin. As a general interest exercise I investigated what effect resistor tolerances would have on the accuracy, and found that in the worst case our sensor would be so inaccurate that it would be completely useless for just about any application. After this I considered complex designs that minimized the errors. I also considered expensive precision resistors. None of these provided acceptable results. Finally I was forced to consider the simplest design and found that the simplest, cheapest design avoided the problem of resistor tolerances and provided accurate results. This experience taught me that theoretically optimal designs are not always optimal once transferred to the real world. It also helped me learn the advantages of considering multiple designs and taught me the skills of evaluating them critically. I am sure that a large part of my engineering career will be spent doing exactly this.

Other than designing the temperature circuit, my contributions to the group included working with the microcontroller. I helped choose the microcontroller and did the initial work figuring out how to program it and communicate with it. The process of reading through a device's documentation was instructive and is another skill that was worth learning.

One thing that made ENSC 440 interesting was that, although it used knowledge gained in other courses, it was nothing like any of the other courses, I have taken. It was difficult to adjust to the fact that we were our own task masters even though we were at school. For the first while it seemed like we were all waiting for assignments from a professor telling us what we should do.

ENSC 440 has provided an experience that will, I am sure, be of great value as I enter the workforce.

6.4 Kevin Ciarniello

The ENSC 440 project has given me a great deal of experience with different things: hardware components, problem solving, time management, stress, documentation, and group dynamics. I thank this experience for really showing me how I can put some of the skills I have obtained in my undergraduate to good use.

I was involved in the hardware selection and design of electrocardiograph circuit, component selection for the temperature circuit, mainly. The other duties I had were working on documentation, the selection of the Bluetooth module and PIC re-programming. Being a 4^{th} year engineer and finally getting a chance to use the knowledge in previous classes, made me feel great.

Some of the things I have gained are troubleshooting a problem that there isn't a cut answer to. I found myself stuck on the ECG circuit and not knowing what to do, and not having clear knowledge of what the answer is, or a professor to go directly to for the answers. I found myself learning to resort to search for the correct result. Before coming into the project, I didn't think of the amount of time that it would take in order to make something. Also, I found it isn't as simple as everyone thinks it is.

I found that it is difficult to work in groups unless there isn't leadership or someone with understand in different areas. I found that our group had a good mix of knowledge in order to get this project complete. It was interesting to find components on the market which sped up the process and at the same time made things more difficult when they wouldn't work.

If I were to do this project again, I would look to start it a little bit earlier and get a majority done before courses start building up. I would also choose components that don't integrate many things together.

The experience of being able to solder onto circuit boards, build schematics and PCB layouts, program in C++, and even step into programming a PIC microcontroller are definitely things I am glad I have learned.

If it wasn't for the messing up of our ToothPIC module, I would've never learned how to troubleshoot PIC processors and this has aided me in expanding my knowledge, which I am grateful for.

The last thing I have found out about this project is that group dynamics are important and that it is important to communicate as a group. There are many factors in developing products and they can't be done alone. Therefore, this is why I feel it is just as vital as building the product, itself.

6.5 Jatinder Singh Mann

I believe the time that we spent on this project was a great learning experience, from both technical and team building points of view. I believe this experience was an invaluable emulation of real life circumstances, and that our whole group will definitely appreciate the lessons learned here.

From a technical perspective, I gained experience in GUI development and integration with hardware. I used Microsoft MFC to develop a real-time animation of the human heart rate, along with other digital readouts. This was good knowledge to pick up, as I have incorporated the animated GUI ideas in my current work term.

From a team building perspective, it was a great experience, as it seemed that we went through all four phases of team building: forming, storming, mourning and performing. Although, one could state that incompetence's of certain members were undesirable, I believe that this was the perfect platform to test such behavior, as the consequences were relatively small.

At the end of the day, we learnt an invaluable lesson of pulling ones weight in the group, and to not wait for instructions, but to push to reach team goals, a lesson that will be easily transferable to the working world.

6.6 Salman Abdollahi

This course was certainly one of my best courses I have taken in engineering. This course taught me many useful skills about team work, time management, and technical skills.

I have learned that in order to achieve positive group dynamics there needs to be clear and healthy communication among group members. At the beginning of the semester we had some problems among members due to poor communication. However, as we got to know each other and understand our limits, we became

more productive working together. I have also learned the importance of leadership in the group.

Time management is extremely important in meeting deadlines with a project of this magnitude. Prioritizing time in order to finish assigned tasks is paramount. However, I have learned that meeting all deadlines is ideal because not everything happens as planned all the time. Also, when time management becomes an issue, managing as a group is sometimes better than managing it individually.

My main contribution to this project was the hardware for the ECG sensor. I have extended my hardware knowledge in general. I have learned about circuit design, different type of op-amps, and filters, which are useful in my engineering career.

In conclusion, I would like to thank everyone in the group for doing a good job on this project. Certainly, it was a unique experience working with everyone and I learned many useful things which are valuable assets for my engineering career.

7 Conclusion

Over the course of this project, the team at Proximus has gone through various stages of research and development. The overall technical skills of the group has increased namely in software development and software integration. Furthermore, the group has gone through several group dynamic issues, but in the end has pulled through to develop the proof-of-concept deliverable described in this document. Through our interaction with medical professionals in the industry, we have gained a greater sense of responsibility and realization on how reliable a medical solution must be before the healthcare industry is willing to undergo a change.

The future looks promising for Proximus, as the WMS is one of the few products in a rapidly developing new industry. Members of the Proximus team will continue to evolve the WMS into a complete solution that will modularize and optimize current healthcare processes in patient care.

8 References

Pictures

[1] http://www.laptop.lt/PICTURES/r3000.jpg

Miscellaneous

[2] Functional Specification for a Wireless Monitoring System," Proximus. February 2006.
[3] Design Specification for a Wireless Monitoring System," Proximus. March 2006.
[4] Microsoft Corporation, Microsoft Knowledge Base Article, http://www.kbalertz.com/Q177101/Modal.Dialogs.Regular.Cause.ASSERT.AfxWndProc.aspx, 2005

