REMOTE MEDICAL INC.

School of Engineering Science
Simon Fraser University
Burnaby, BC, V5A 1S6
ensc440-rabbit@sfu.ca

April 8th, 2005

Lakshman One
School of Engineering Science
Simon Fraser University
Burnaby BC
V5A 1S6

RE: ENSC 440 Project Remote Health Monitor Design Specification

Dear Lucky,

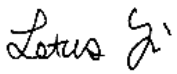
We have attached the Remote Health Monitor Post Mortem based on the project described in the proposal we submitted previously. Remote Health Monitor is to collect a patient's health data and manage them in a low cost, scaleable, and user-friendly fashion. Patients can be monitored anywhere there is the internet access. The Remote Health Monitor measures health parameters such as body temperature and pulse, and then transmitted the readings to the hospital's database through the internet. With the software that comes with the package installed, patients can also access their personal clinical information.

The purpose of the following document is to describe the current state of system, differences from our original design specifications, and recommendations on future plans with the CAPIS system. Also included in this document are outlines of our time constraints, budget, and the interpersonal and technical experience gained by each individual through working on this project.

The post mortem document describes in detail the difference between our original designs from the final implementations of each module of the remote health monitoring system. Problems encountered, budget comparison, team dynamics as well as future expansion, are discussed.

Remote Medical Inc. consists of four experienced and enthusiastic 3rd and 4th year engineering students: Dong Zhang, Calvin Che, Marian Chang, and Lotus Yi. If you have any questions or concerns, please feel free to contact us through the email at ensc440-rabbit@sfu.ca.
Sincerely,

Sincerely,



Lotus Yi
Chief Executive Officer
Remote Medical Inc.

Enclosure: *Post Mortem for Remote Health Monitor*

REMOTE MEDICAL INC.

Post Mortem for
Remote Health Monitor

Project Team:

Marian Chang
Calvin Che
Lotus Yi
Dong Zhang

Contact Person:

Lotus Yi
ensc440-rabbit@sfu.ca

Submitted to:

Lucky One – ENSC 440
Mike Sjoerdsma – ENSC 305
School of Engineering
Simon Fraser University

Issued Date:

April 8, 2005

ABSTRACT

Management of patients with chronic conditions is a long-standing challenge for health care organizations. These conditions include diabetes, chronic heart failure, asthma, HIV / AIDS, and cancer. Patients are required to adopt lifelong diet and drug control to maintain optimal health and avoid the complications of the disease. These complications can arise suddenly and can be life threatening. Therefore, the health condition of patients with chronic diseases should be reported frequently.

We propose to implement the data reporting into a single stationary device that transmits health information for physicians via internet connections. Our solution, Remote Health Monitor (RHM), measures a patient's medical conditions and sends the measurements to the hospital database through internet. The design of a stationary device ensures consistency of the environment under which measurements are taken. The database can be accessed either by the physicians or patients with specific login identifications to keep information confidential.

This post mortem describes our group dynamics, comparison of our estimated and actual budget, comparison of our proposed timeline and the real schedule, individual contribution to the project, problems encountered, future plans for the system, and things we learned while implementing our product.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

A / D converter	Analog to digital converter, convert an analog signal in voltage levels, into digital signals in bits.
API	Application Program Interface, the interface application programs use for accessing services provided by some lower-level module.
ASCII	American Standard Code for Information Interchange, a standard character-coding scheme that assign one bit pattern for the lower 7-bits to one of the 128 different characters.
bits	Fundamental of digital information with possible values of 0 and 1.
bpm	Beats per minute, a measurement of pulse.
bytes	A collection of 8 bits
CSV	Comma Separated Values, a format that can be used as a portable representation of a database, and can be imported into other databases or popular spreadsheet programs such as Microsoft Excel.
HTML	Hyper-Text Markup language, a document format language that is used in the World Wide Web and can be opened using a web browser.
interrupt	The suspension of normal program execution to perform a higher priority task as requested by a peripheral device. After completion of the task, the execution of the interrupted program is resumes from the point where it was left off.
I / O	Input and output ports, which is used along with connection cable to link one device to the other and allow communication.
IP	Internet protocol, the basic network transmission protocol of the Internet.
IP address	An address with 4 numbers separate by periods that uniquely identifies a certain computer on the internet.
key	The column in a database table.
LED	Light-emitting diode, a semiconductor device that produces light when conducting currents.
MySQL	An open-source SQL database developed at www.mysql.com. It is common used for small business.
protocol	Standard procedure for regulating data transmission.

register	A memory device that has a specific address and is used to hold temporary data.
Rabbit 2000	An 8-bit microprocessor that is made by Rabbit Semiconductor Inc. and is commonly used in embedded systems.
RHM	Remote Health Monitor, the name of our product.
RTF	Rich text format, a method of encoding text formatting such as font and colour using ASCII text.
SPI	Serial peripheral interface, a method of communication between host processor and peripheral device with data sending one bit at a time through a single channel.
SQL	Structured Query Language, a standard interactive and programming language for getting information from and updating a database.
TCP / IP	Transmission control protocol / internet protocol, a protocol that computer used to communication and exchange information over the internet.

1. INTRODUCTION

The goal of the Remote Health Monitor (RHM) is to collect a patient's health data and manage them. Patients can be monitored from anywhere there is internet access. The RHM measures parameters such as temperature and pulse, and transmits the information to the database located in the hospital. The database can be accessed through a software application that will be provided along with the purchase of RHM. The application can be logged in to either as a doctor or as a patient, thus allowing the care providers to monitor a patient's health condition and the patients to review their own medical history.

Our goal in implementing RHM is to provide chronic disease patients with an easy to use device that will help them track health information without going to the hospital, and to allow doctors to review the patient data with a software application that organizes and updates information automatically.

This document details the various issues we encountered, and problems we solved during the implementation of RHM. Also, each group member's contributions are acknowledged. Finally, by comparing the estimated and the actual timeline and budget, we can gain more insight of how we should perform if we take on a similar project in the future.

2. CURRENT STATE OF THE SYSTEM

Figure 1 shows the high level structure of the system. The database server is located in the hospital; the Microcontroller – Rabbit 2000, and measuring devices are installed in the patient's house; the user can access from any PC with the client software installed and internet access, and can log in either in doctor or patient mode.

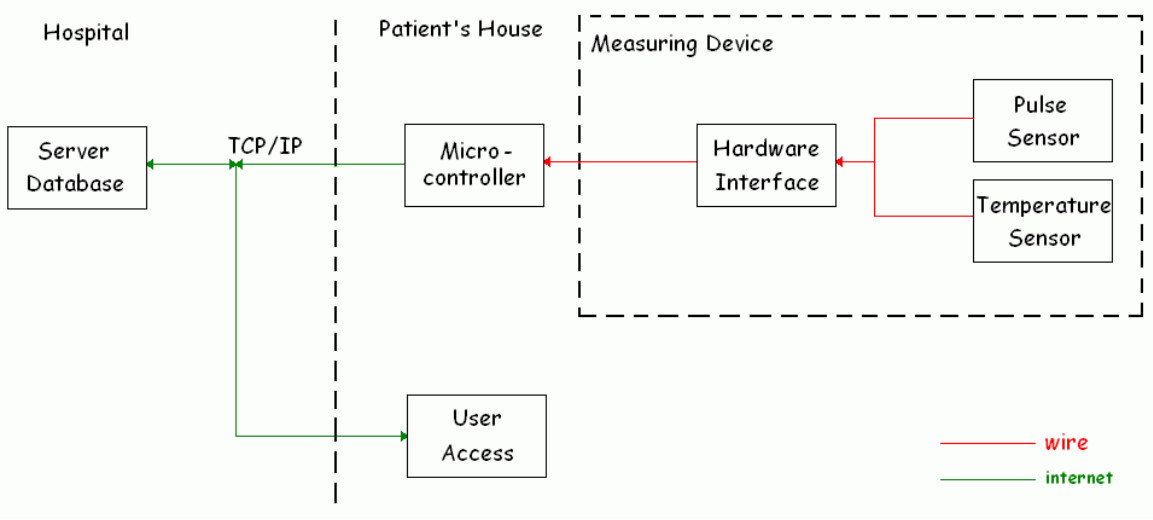


Figure 1: System Block Diagram

Figure 2 presents the basic functionality of RHM in a graphical manner.

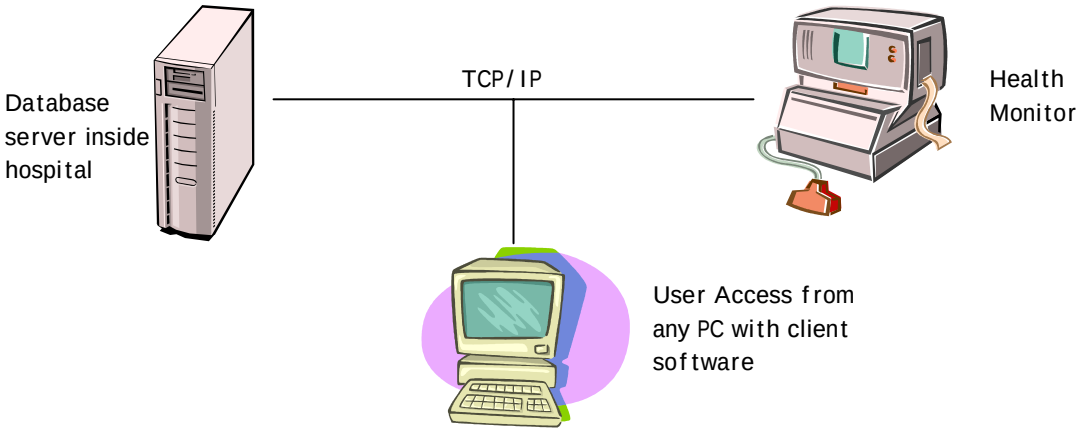


Figure 2: Graphical Diagram of the System

The current state of the system for each of the hardware, firmware, software, and database is described in detail in the following sections.

2.1 Hardware Component

In determining the technology to be used for body temperature and pulse measurements, we seek solutions that are non-invasive to minimize the discomfort when the measurements are taken. Both of our sensors acquire data optically.

2.1.1 Temperature Sensor

For the temperature sensor, we utilize the IR sensing technology to detect the black body radiation from a person's ear, and translate the amount of radiation detected to the body temperature. For our prototype, we place the sensor inside an old mouse, and upon measuring the temperature, the user place the opening at their ear for temperature acquisition. With the IR sensing technology, we can measure the body temperature accurately and non-invasively. The accuracy of the sensor is 0.2°C.

2.1.2 Pulse Sensor

For pulse measurements, we use a reflective sensor to sense the blood flow at the finger tip of the patient. This method utilizes the idea of oximetry, a method of monitoring the percentage of haemoglobin (Hb) that is saturated with oxygen. To measure pulse, we implement a single wavelength IR diode, along with a photo transistor to detect the difference in the amount of blood at the finger tip as the heart beats. Upon taking measurement, patients need to place their finger at the designated spot covering the opening of the reflective sensor. The accuracy of the pulse sensor is 5 bpm.

2.2 Firmware Component

2.2.1 User Interface Handler

User interface handler is used to efficiently and correctly obtain user's health data. We connect an interface board, shown in Figure 3, which consists of four push buttons, four LEDs, and a buzzer to the I/O of the Rabbit 2000. The first set of push button (SW1) and LED (LED1) corresponds to pulse measurements, and the second set of push button and LED corresponds to temperature measurements. After the user set up the monitoring device, the push button must be pressed to indicate the start of measurements. A beep sound indicates the data is successfully obtained and transmitted. If the data obtained is not within a reasonable range, the corresponding LED would flash to indicate a non-successful data acquirement.

User interface handler is implemented using the two external interrupts, one for pulse measurement and the other one for temperature measurement. Two similar interrupt service routines are designed for each of the measurement.

Figure 3: The Demo Board of Rabbit 2000

2.2.2 Temperature Measurement

For the temperature, we utilize the parallel communication. The output voltage of the IR sensor is between 0V and 4.5V. The relationship of the output voltage to the temperature of the object measured is

$$T = 0.61\ V_{out} - 20,$$

where T is the measured temperature and V_{out} is the analog output voltage on TEMPOUT pins. To obtain the digital output, we use an 8-bit analog to digital converter. The output is then connected to the parallel port of the controller for further processing.

2.2.3 Pulse Measurement

For the pulse measurement, we utilize the on-chip input capture channel, which simplifies the solution to software only. Basically, after proper setup, the input capture channel will wait for certain action that is happening on an input pin. There are two conditions defined for the use for the input capture channel: starting condition and ending condition. When the starting condition occurs, the timer is started; after the ending condition is caught by the processor, the time difference between start and end is saved in one register. For our application, since we are measuring the pulse rate, the duration of the square pulse will give us the duration of the pulse, provided that the pulse sensor will produce a square pulse in accordance to the patient's heart beat, namely, a rising edge for one beat of heart. In our situation, the starting condition and ending condition should both be the rising edge of one pulse. Thus, the time difference will give us the duration for the pulse, from that the heart beat rate can be calculated.

2.2.4 Communication Through the Internet

To use reliable transport services, TCP hosts must establish a connection-oriented session with one another. Connection establishment is performed by using a "three-way

handshake" mechanism. For our project, we use the server-client mode to implement the data transmission. After the measurement is done, Rabbit 2000, the microcontroller, will initiate a connection to the Database Server and start the transferring of data. On the Database end, we implement a server written in C#. The server keeps listening to a specific port and waits for the initiation of the connection. After a connection is successfully established, the server will process the data sent by Rabbit 2000 and store the data into the database.

2.3 Software Component

Our client software, MediNet, is written using the C language and compiled using Microsoft Visual Studio. It allows both doctors and patients to access to database and view the measurement data.

2.3.1 Access Mode and Key Features

From the user ID, the program can determine whether the user is a doctor or a patient, and grant the access to the database accordingly. Doctors can view the list of all monitoring patients, and the personal information and measurement data for each of the patients. They can also view and change his information, as well as the machine association for the patient. Patients can view their own measurement data as well as the personal information for all their doctors. Also, they can also view and change their own information as well as the machine association. Other key features for the software include password encryption, configurable server address, patient status summary, monitoring list management for doctor with function such as add and remove patients, and multi-user features.

2.3.2 User Interface

The interface of the software is similar to many other applications running under Windows, which include a menu bar, a toolbar, a workspace, and a status bar. A screenshot of the main window is shown in Figure 4 on the following page. The menu bar contains access for all functions of the program while the toolbar contains the key features; the workspace displays a list of information or data so that the user can view and / or modify; the status bar tells user the current state of the program.

2.3.3 Multi-User Feature

For our implementation of the device, the RHM are hardcoded with a machine ID but not with any other data. For writing measurement data into database, we store an association between patient ID and machine ID into the database, so the patient ID can be retrieved by comparing the machine ID received from the microcontroller with the ones stored in the database. To allow multiple users to use a single machine, we implement an algorithm that can change the association between a patient and a machine.

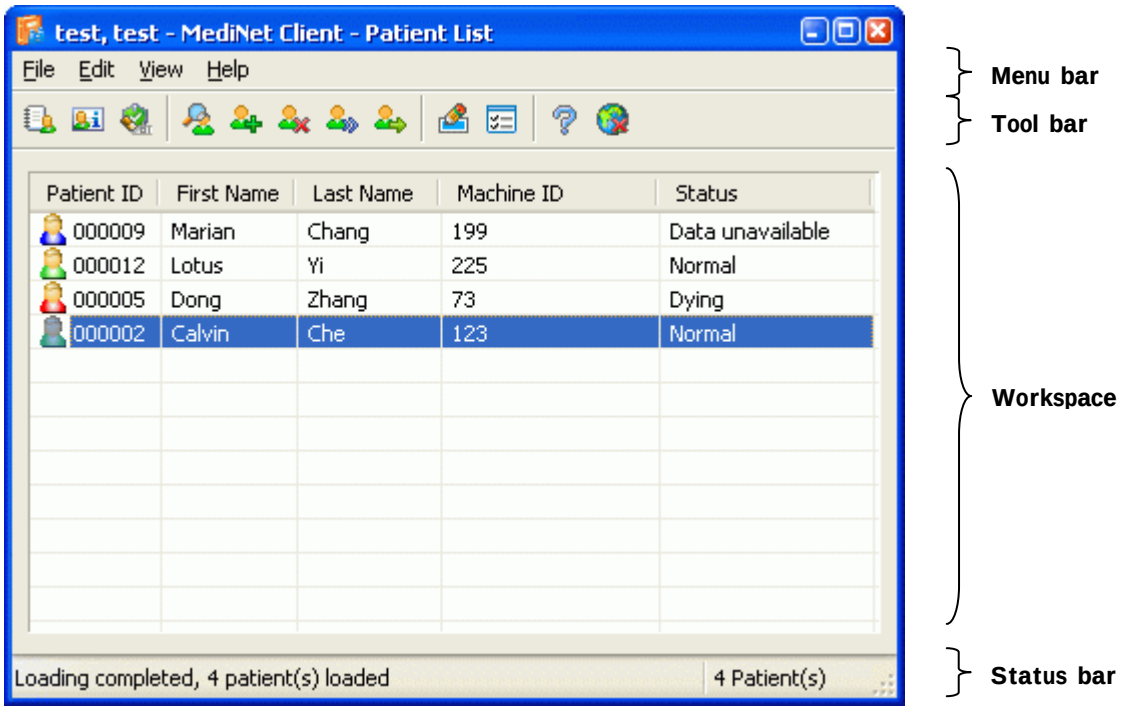


Figure 4: Main Interface for MediNet Software

To change this association, a direct connection between the client software and microcontroller board is required, and is implemented using the double handshake algorithm. This connection is made after user enters the machine's IP address to obtain machine ID, and it also serves as a way to prevent user from associating himself or herself to the wrong machine and leading to loss of data. After the machine ID is obtained, the software then write it into the database. The machine and patient is a 1-to-1 association, which means that once a new association is created, the old one will be removed.

2.4 Database Component

The database is implemented with MySQL 4.0.13 using 4 tables. The first table "detail" stores the user information such as ID, password, name, and contact information. The tables "temperature" and "pulse" are used to store the measurement data for all users, one for each type. Finally, the table "assoc" stores the patient to doctor association, which is essential for implementing the many-to-many association as outlined from the specification. All access to and from the database, from both client software and the server application, are using the API provided by MySQL.

3. DERIVATION FROM PROPOSED PROJECT

3.1 Schedule Derivation

The estimated timeline produced at the beginning of the semester is shown in Appendix A at the end of this document. The actual timeline for our project, shown in Appendix B, is quite different from the schedule we expected. The implementation phase for both hardware and firmware is much longer than planned. The major issue is the selection of sensor type. Originally, we chose something that is really hard for us to do, so a lot of time is wasted, and we ended up having to switch to alternative solutions. For the temperature sensor, we planned to use the one with a SPI interface, but the sensor we purchased somehow has a different communication protocol than the software library provided by the development board vendor. After weeks of trying, we finally decided to use an A/D converter in combination of the original sensor analog output to solve this problem. The story is kind of the same for the pulse sensor; the first design of the pulse sensor is to amplify the sound of the heartbeat, but due to hardware limitation we were not able to get a strong pulse signal just by using a microphone and amplification devices. The second solution is an IR phototransistor sensor, which works but is not very stable in different ambient light conditions, which resulted in a huge amount of debugging and testing time. These are the two main reasons for the big deviation from the original schedule.

For the software and firmware component, there is a change from the original plan. Rather than writing the entire software then test and debug it as a whole, we decide to test the program as we are writing it. This way, all errors can be found and fixed soon after they are written, and prevent it from being propagated or hidden as the program become large. This method will save time for testing the program.

3.2 Budget Derivation

Table 1 compares the estimated and the actual cost of our project. We have over estimated the amount of money we need to implement the pulse sensor because we use a different sensing method, oximetry, instead of the originally proposed pressure sensing technique. For the temperature sensor, the cause of under estimation is due to extra shipping cost and cross-boarder examination fee that were not taken into account at the beginning. When estimating the cost, we thought we need to use special shielded cable in some parts of our design to avoid interference; however, by utilizing thrown-away parts found in the lab, we save money on the extra cables, wires, and case. Though we have some under-estimates, but the over-estimates compensate those amounts, and allow us to achieve realistic overall budget estimation.

Table 1: Comparison between Estimated and Actual Cost

Components	Estimated Price	Actual Price
Rabbit2000 TCP/IP Development Kit	$250.00	$250.00
Pulse Sensor	$80.00	$50.00
Temperature Sensor	$60.00	$200.00
Cable/Wires	$25.00	$0.00
Case	$30.00	$15.00
USB Serial converter	$50.00	$30.00
Total Cost (10% contingency)	**$550.00**	**$545.00**

We have also calculated the cost to produce our system without paying for research and backup parts. The total cost required is $150, and the details are summarized in table 2.

Table 2: Break Down of the Production Cost

Components	Cost
Rabbit2000 TCP/IP Module	$35.00
Pulse Sensor	$10.00
Temperature Sensor	$90.00
Case	$15.00
Total Cost	**$150.00**

4. PROBLEM ENCOUNTERED

4.1 Hardware Problems

Pulse measurement through devices was a challenge to us, since we usually place our fingers at our wrists to measure the pulses. Besides the optical solution we implemented as the final design, we have also looked into measurements utilizing pressure sensors and microphones, but they do not work as well as the optical solution. For the pressure sensor, we were unable to find a sensor can measure the fluctuation of our skin without pumping gas into the sensor. For the microphone, without proper amplification at the data acquisition site we could not hear the heart beat. However, the circuits we designed while implementing these solutions were useful in implementing the optical solution.

4.2 Firmware Problems

To obtain the temperature reading, we originally utilized the serial communication protocol, SPI, to communicate with the infrared thermometer module because the thermometer module has a built-in digital interface that is SPI compatible. At first, we could not get any output from the module. After a period of debugging, we realized that the timing requirements were not satisfied. We made some modifications, and were able to get the output temperature value from the module. However this value stayed constant and did not change with the temperature fluctuation. The information given in the datasheet was very limited, so we decided to contact the company. However, the technical support still was not able to resolve the problem. Because the time constraint, we abandoned this mechanism and get the temperature value from the module using the analog output and an A/D converter.

4.3 Software and Database Problems

The main problem encountered in this part is the design of database for storing and retrieving data efficiently. One particular problem is for the storing of the doctor and patient association. From the design, this needs to be a many-to-many association, which means that each doctor should be able to monitor unlimited amount of patients and each patient could be monitored by unlimited amount of doctors. The original design is to have 5 keys in the “detail” table that store the monitoring doctors for each patient. Although this approach is able to have the doctors monitoring unlimited amount of patients; however, each patient can have only up to 5 monitoring doctors. This may be enough for most situations, but if the hospital wants to set up some accounts to monitor all or specific groups of patients, this solution will fail. The final solution is to create another table that stores each of the association as an entry. Because a table can have unlimited entries, the problem is solved. A consequence for this implementation is the increase of network traffic and processing time as the lookup for the association now requires 2 database queries instead of just 1 as originally planned.

Another problem we found is the implementation of the multi-user design. For adding such feature into our product, the unique identifier for the current user cannot be hardcoded into the machine because it must be changeable. Our initial plan is to have the client software send the user ID to the microcontroller, and the microcontroller will write the measurement data into the database using that ID. However, because there is no way to have the board remember the information after power down, we either have to let the user send their ID to the machine at every startup or think of an alternative solution. Letting user to send their ID every time defeats the purpose of being remote, especially for the case when there is only one user for the machine; therefore, we decided to have a machine ID and implement the association of patient ID and machine ID in the database.

5. FUTURE EXPANSION

There is no final product in the high tech industry, extended research and development should be done as soon as a new product is introduced to ensure the compatibility in the market. We plan the same. The following sections outline some possible expansions for our product in the hardware and firmware, software and database categories.

5.1 Hardware and Firmware Expansion

Since our hardware and firmware is highly portable, we are thinking of several enhancements to our design, such as adding other useful measurement devices to our system. Blood pressure and glucose level measurements are also very useful in determining the health condition of the users. Also, for the pulse sensor, we can modify it such that measuring the percentage of haemoglobin would be possible.

An extra feature that makes the device more user-friendly is to add a LCD display. The LCD display can tell user the current state of the machine, such as the current user and the current measurement data. It can also warn user if a specific error has occurred, rather than flashing LED which may be ambiguous sometime.

Another possible extra feature will be to include a camera and audio system, so the patient and doctor can talk to each other if required. This of course will require a more powerful processor and other more expensive equipments like camera, microphones and intermediate signal processing systems. And the cost of the unit will naturally increase.

Currently our product uses wired Ethernet connection; it would be more convenient if wireless solutions are provided. This can be in the form of wireless Internet connection, or just custom made wireless communication systems. The former one is more mature and has many available modules on the market, but it requires the user to pay for the connection, which adds a running cost to our product. The latter solution does not require this cost, but needs to be implemented from scratch, since no modules are available in the market, the research and development cost will increase dramatically, at the same time, the unit cost will go up too. For now, we are still researching possible add-on features depending on the need of the user.

5.2 Software and Database Expansion

An improvement to the existing software is to add more supported formats for the export function. The current program can only export the data into a plain text document, which is hard to read and analyze. Other possible formats include HTML, RTF, and CSV. HTML and RTF formats enables a better formatting of data by using table, and the different levels of data such as warning and critical can be represented using a different colour of

text. CSV document enables the transport of data into a different database, or import to spreadsheet software such as Microsoft Excel for ease of analyze.

Another possible addition to the client application is to create graphs and figures using the data. This, for example, can be a line graph showing the trends of most recent data, a pie graph showing a relative amount of normal, warning, and critical data in the recent days, or a bar graph that shows the amount of data in each level for each month. These graphs present the data in a way that is easier to read and understand, and can let to a better understanding for the patient about their health condition.

A possible feature to be added to the database server is to automatically send notification to patients and doctors if a critical measurement data is found. This notification can be in the form of e-mail or phone (the e-mail address and phone number can be obtained from user's personal information). To implement this feature, the easiest way is to modify the server application that will monitor the data as they are coming in. Also, this feature will involve some additional options for the users such as to enable or disable the notification. The settings can be stored in the database and they can be changed using the client software, MediNet.

Finally, an improvement for sending and receiving data to and from the database is the implementation of secure data transmission. This can be done by adding a verification code to each data transmission, so that fake data can be rejected by the server. The data itself can also be encoded so that there will be no leak of personal information if the data is hijacked during the transmission.

6. GROUP DYNAMICS AND TECHNICAL EXPERIENCES

The project is divided into three main different areas. Lotus and Dong are responsible for the firmware part, Marian for the hardware, and Calvin for the software and database. Lotus is the CEO of our team. Each week, one meeting was organized, and the agenda is set by Lotus. In the meeting, each of us reported the progress made and problems encountered, and then we discussed the solutions and possible approaches together. Marian is responsible for the meeting minute. After each meeting, a summary was sent. This helped us organize the project and keep each of us on track. We had a very good time working together, and we were able to complete the project on schedule. The technical experiences learned by each member are described in the following sections.

6.1 Experiences from Marian Chang

I am responsible for the hardware components of the system. My responsibility includes researching on methods that could be used to achieve desirable measurements, and implementing the solution selected. Through research and implementation, I learned various methods of data acquisition, and different design of hardware filtering and their impact on the signal.

If I were to undertake a similar project, I would limit the time on the research level, and start experimenting with different solutions at an earlier stage of the project cycle. For the hardware implementation, factors such as ambient noise and interference are hard to predict with theory alone. Also, a solution to the problem that seems difficult to implement at first may end up being the most effective and efficient solution to the problem. Therefore, I think research should be done simultaneously while experimenting with different solutions to gain more insight of the design, and to acquire a more practical view their feasibility.

6.2 Experiences from Lotus Yi

I am responsible for the firmware and database server components of the system. First of all, I researched on an appropriate microcontroller that would best fit the purpose of the Remote Health Monitor. I chose the Rabbit 2000 TCP/IP development board and then went on to investigate on how to program the controller in C language. I also spent some time understanding different functionalities of the microcontroller. Then I picked the suitable mechanisms for pulse and temperature measurements as well as the user interface. I am also involved the software development for the database server. For this project, I chose the microcontroller without fully listing all the features and functions needed for the application. Therefore, I have some constraints when designing. If I were to do the project again I would think through all the functions and features carefully before choosing a suitable microcontroller.

6.3 Experiences from Calvin Che

I am responsible for the software and database components of the system, and help Lotus on the database server regarding the database access. My responsibility includes the selection of the database software to use, design of the database and table structures, research on methods of accessing database, and design and implements the client software program that will fulfill the requirements set by the specifications. The main challenge for the database part is to design the tables so it can be accessed efficiently, and the main challenge for the software part is to make sure it is bug-free. At times there are crashes in the software due to memory corruption, and a lot of time is spent to fix these issues.

If in the future I am involved in a similar project, I have to pay more attention on allocating and freeing memories in order to prevent the memory corruption errors. Also, I can use the insight and tricks I learned about the database structure to facilitate the design process of a future database.

6.4 Experiences from Dong Zhang

I am responsible for the implementation of the firmware. My responsibilities include development of the firmware according to the requirement of the server application and hardware specification, debugging and interfacing with the hardware. The major challenge of the work is to decide the source of error whenever the system is not functioning desirably because of the limitation of the development kit, especially with the hardware. In which case, I have to figure out what the problem is by eliminating and isolating various factors.

If in the future I am involved in a similar project, I would first choose a development tool that has convenient and user friendly debugging features; however, this is not likely the case, since most of the development tool are custom made by the supplier, specifically designed for the system that we have selected. Another lesson that I learnt is that as design engineers, we should not think like an engineer when it comes to the design of user interface, user can not think like designers, so it is not fair to force the user to comply the way of operation that we think is convenient. Always, we have to put ourselves in the shoes of the user to make sure our product is user friendly.

7. CONCLUSION

The development process of this project is a very valuable experience in the sense that we leant to collaborate in a team, thinking and cooperating to solve problems at hand, manage timeline and maximize efficiency to meet deadline, be suggestive and compromise at the same time to adapt alternative approaches, and yet keeping a healthy relationship within the team. This project gave us a feeling of the research and development work in the industry, which likely in the future we will invest our lives in. The accomplishment is not just a product that works, but a lesson, an enlightenment, and a touch of the real world that prepares us for the actual career that we will pursue in the near future.

8. SOURCES AND REFERENCES

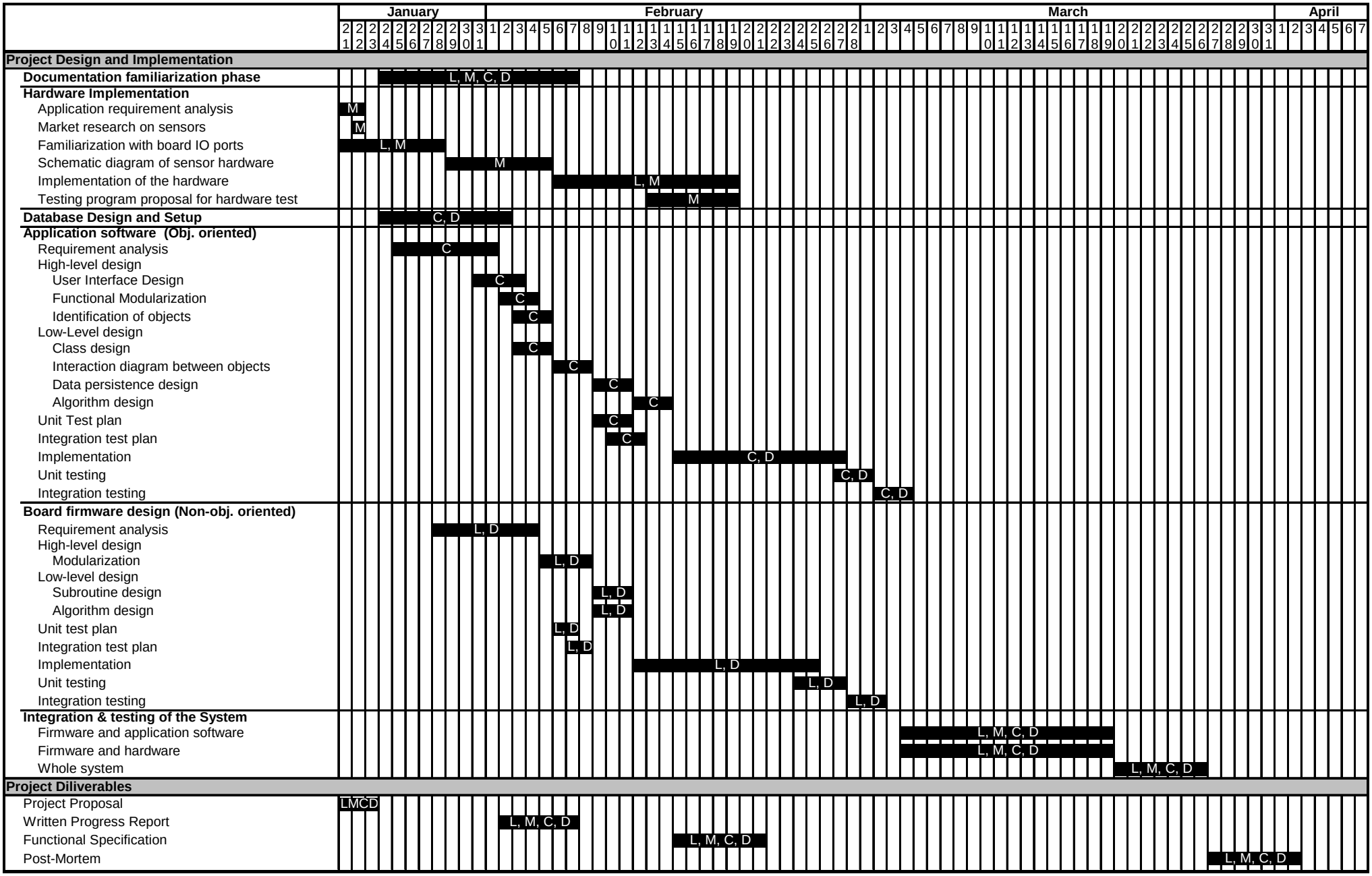
[1] Rabbit Semiconductor, “Rabbit 2000 TCP/IP Development Kit,” 2005, http://www.rabbitsemiconductor.com/products/rab2000_tcpip/index.shtml.

[2] Melexis, “MLX90601 family – IR thermometer modules”, 2002, http://www.melexis.com/prodfiles/mlx90601.pdf.

[3] EMKAY, “MD9745APA-1 Product Specification”, http://rocky.digikey.com/WebLib/Emkay%20Products/Web%20Data/MD9745APA-1.PDF.

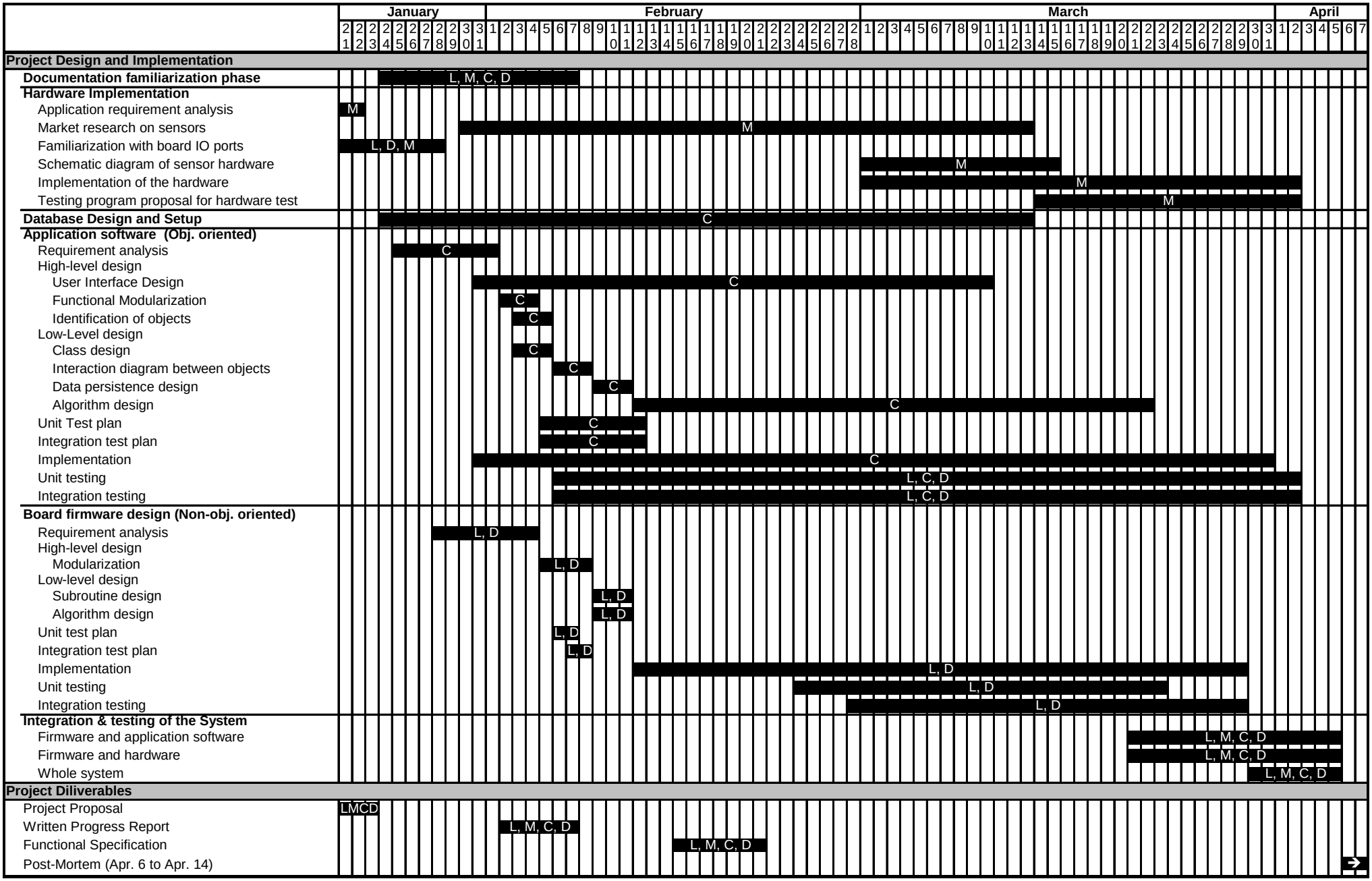
[4] MySQL Reference Manual, 2005, http://dev.mysql.com/doc/mysql/en/index.html.

APPENDIX A ORIGINAL PROJECT SCHEDULE



Legend: L - Lotus Yi M - Marian Chang C - Calvin Che D - Dong Zhang

APPENDIX B ACTUAL PROJECT SCHEDULE



Legend: L - Lotus Yi M - Marian Chang C - Calvin Che D - Dong Zhang