Intueor Systems

April 19, 2007

Mr. Steve Whitmore
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

Re: Post-Mortem Report for the PowerMoleDistributed Power Monitoring System.

Dear Mr. Whitmore: Enclosed is the *Post Mortem Report for the PowerMole Distributed Power Monitoring System* and it outlines the process our team went through when designing and implementing our project for ENSC 440. Our objective was to build a device that would allow the remote monitoring of the power consumption of an average household device.

This document details the current state of the device, deviations from our original plans, and our future plans for the device. In addition, we outline some of the budgetary and time constraints we encountered and explain the inter-personal and technical experience gained from working on this project.

Intueor Systems is the creation of three senior engineering students: Daniel Ralph, Dimitri Tcaciuc and Adam Leszczynski. Feel free to contact me any time at 604.585.4606 or by email at dralph@sfu.ca.

Sincerely,

Daniel Ralph
Chief Executive Officer
Intueor Systems

Encl: *Post Mortem Report for the PowerMole Distributed Power Monitoring System*

Post-Mortem Report for the PowerMole Distributed Power Monitoring System

April 20, 2007

Intueor Systems

PowerMole™

Dan Ralph {dralph@sfu.ca}
Adam Leszczynski {aal@sfu.ca}
Dimitri Tcaciuc {dtcaciuc@sfu.ca}

Contents

List of Figures

List of Tables

List of Acronyms

AC: Alternating Current

ADC: Analog to Digital Converter

ASL: Application Support Layer

API: Application Programming Interface

LCD: Liquid Crystal Display

MAC: Media Access Control

MLME: MAC Layer Management Entity

NLME: Network Layer Management Entity

NWK: Network Layer

OOS: Open Source Software

PAN: Personal Area Network

PC: Personal Computer

RRD: Round Robin Database

UART: Universal Asynchronous Receiver Transmitter

USB: Universal Serial Bus

ZDO: ZigBee Device Objects

1 Introduction

For the past four months, Intueor Systems has been actively designing implementing a proof-of-concept prototype of the proposed PowerMole Distributed Power Monitoring System. It is intended to monitor the power consumption of an arbitrary appliance and then wirelessly transmit the data to the PowerBurrow, a central monitoring station that aggregates and displays statistics. This will allow companies to better understand how much energy the multitude of electronic devices that they employ consume. This allows them to become more "green" in a environmentally conscious world as well as make more informed choices about where to deploy new appliances. The first stage of the PowerMole Power Monitoring System, a proof of concept, has been completed.

In this document, we discuss the current state of the prototype, provide future plans and recommendations, and assess the development of the prototype from a project management standpoint. Furthermore, each team member from Intueor Systems offers insights and reflections on the inter-personal and technical experiences gained from collaborating on this project over the past few months.

2 System Overview

Figure 1 gives a conceptual overview of the PowerMole power monitoring system.

This system contains one or more remote monitoring devices which wirelessly transmit data back to the central monitoring station. These transmissions can be done at regular intervals or upon request. Depending on the versatility of the central monitoring station, the user may request a statistical data analysis, set device alert conditions and more.

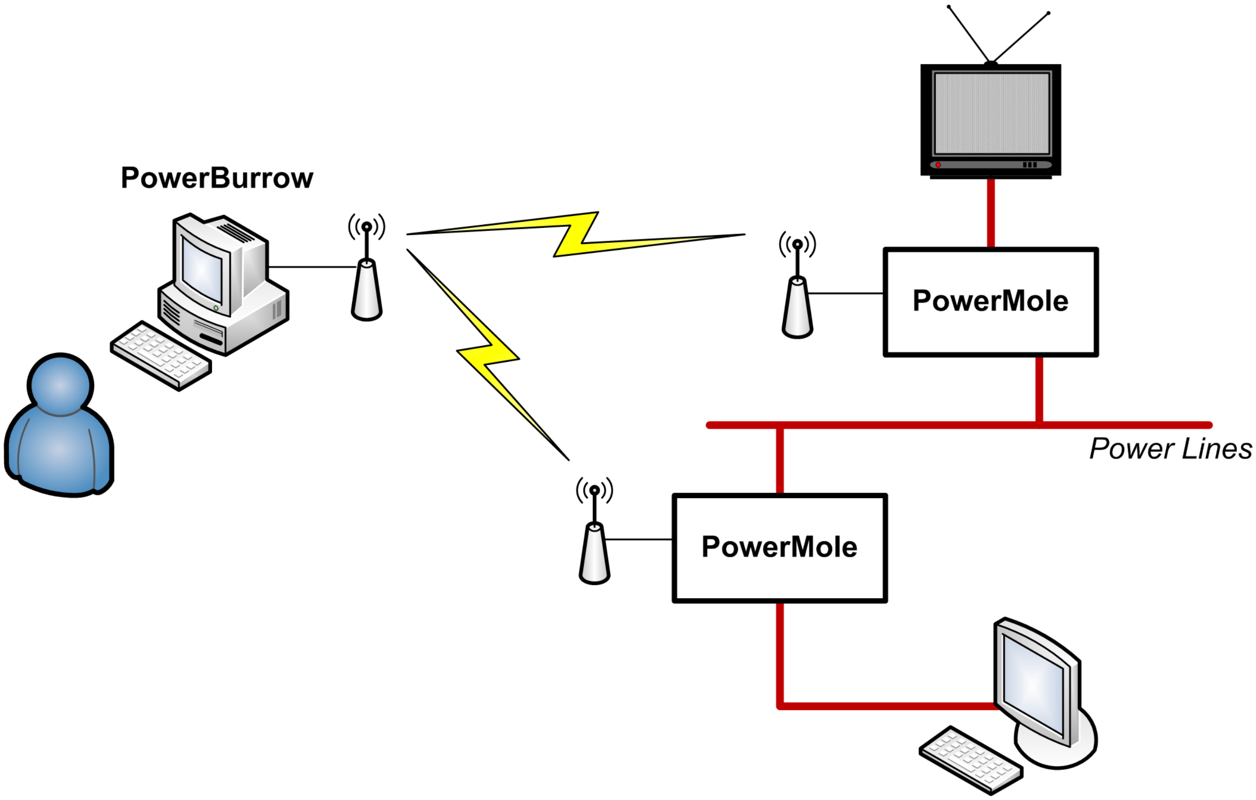


Figure 1: Distributed Power Monitoring System Overview

3 The PowerMole Prototype

This section discusses the current state of the PowerMole modules. Figure 2 gives a top level overview of the major components.

3.1 Atmel AT Mega32L Microcontroller

The Mega32 is used to control the overall operation of the PowerMole. It periodically collects data from the current meter, stores this data internally, and controls operation of the ZigBee module to wirelessly transmit information. Once every second the Mega32 wakes up, turns on the Analog to Digital Converter (ADC) and goes to sleep until the ADC has completed a conversion. The data is stored and the Mega32 checks to see if the PowerMole is full and if it is the PowerMole attemps to send the data to the PowerBurrow and then goes back to sleep. Because the PowerMole spends most of its time sleeping, it conserves power.

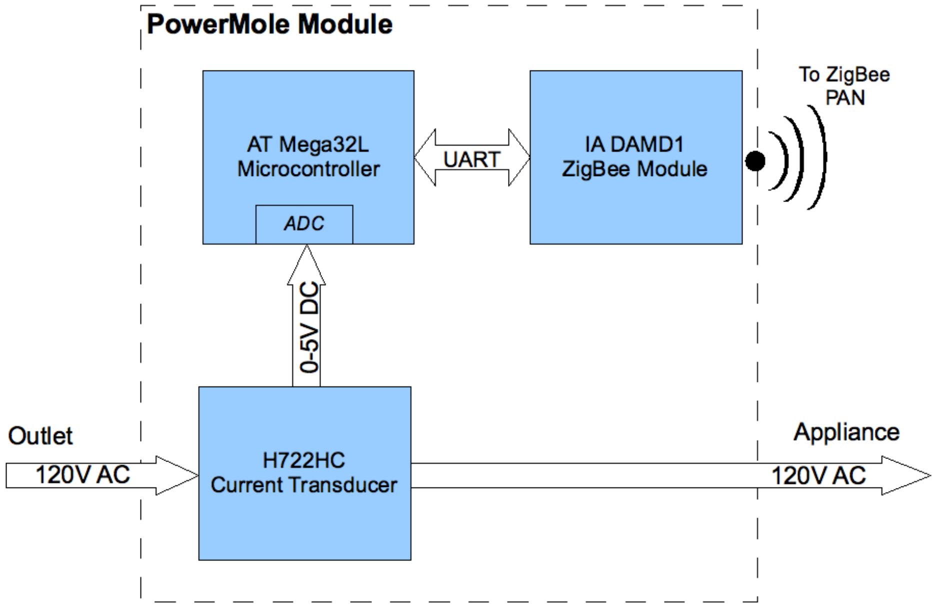


Figure 2: PowerMole Hardware Block Diagram

3.2 IA OEM-DAMD1 2400 ZigBee Modules

The IA OEM-DAMD1 2400 was purchased from Integration Associates, Inc.. The DAMD1 is controlled by the Mega32L microntroller through a serial interface and is used for wireless transmission to the central monitoring system. For the central monitoring system, a ZigBee USB dongle was purchased. The dongle contains an identical ZigBee module with a USB interface for easy integration with a PC.

3.2.1 Hardware Design

The ZigBee protocol, in general, was chosen to meet both the wireless connectivity and power requirements outlined in the functional specifications. ZigBee was chosen over other wireless protocols, such as Bluetooth and WiFi, because of its low power consumption and compartively large range. The DAMD1 device has a typical power rating of 1mW and can transmit over 60m. In comparison, 1mW Bluetooth devices have a typical range of only 1m, and the lowest power WiFi units are rated at about 200mW.

The DAMD1 device, in particular, was chosen due to its affordability and ease of use. In addition to the physical and Media Access Control (MAC) Layer, defined by the IEEE specification, these modules also provide the Network (NWK) Layer, the Application Support Layer (ASL) and the Zigbee Device Objects (ZDO) Layer. This software allows designers to focus on application programming without worrying about the underlying wireless specification. Figure 3 shows an overview of the ZigBee stack provided in the DAMD1.

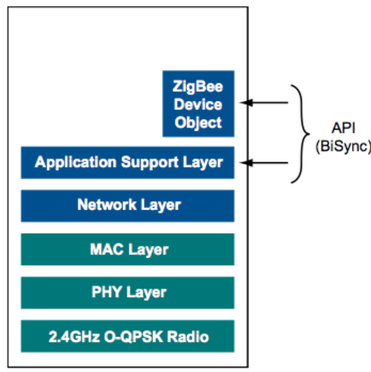


Figure 3: ZigBee Stack

Source: www.Integration.com, IA OEM-DSMD1 2400 datasheet

3.2.2 Software Design

At the software level, the ZigBee modules must be able to establish and communicate over a Personal Area Network (PAN). According to the ZigBee specification, the device that creates a PAN becomes the 'coordinator' while other devices, known as 'children', join the network. In this scenario, the central monitoring system will act as the coordinator while each PowerMole unit will join the network as a child.

3.3 Veris H722HC Current Transducer

The H722HC current transducer is used to measure the current draw of connected appliances. The current running through the solid core is converted into a DC voltage, which is then read by the Mega32L through an onboard ADC. The H722C outputs a voltage of 0-5V DC, based on a current of 0-50 amps. The ADC converts the value given to it into a 10 bit unsigned number with 0x3FF being equal to 2.56V. Even though the H722HC can output a much greater voltage, the average household device will not need this current.

4 The PowerBurrow Prototype

The PowerBurrow is the central monitoring and data storage station. It is a software package as well as a USB ZigBee dongle to allow it to communicate with the PowerMoles. Because of this, there was no hardware that needed to be designed. This section will deal with the current state of software that composes the PowerBurrow. The PowerBurrow is broken down into several components and these are: Device Information, Power Consumption Storage, Data Acquisition and User Interface.

4.1 Data Acquisition

Data is aquired wirelessly using a ZigBee Dongle. There are two ways that data gets to the PowerBurrow: the PowerMole fills its data storage and sends the data over to the PowerBurrow, or a user requests for data to be sent over.

4.2 Data Storage

Data storage is split up into two main components: the information about the device and the data that the PowerMole has collected.

4.2.1 Device Information

There are several pieces of information that need to be stored about the monitored device, such as a description of the device, the periodicity as well a system ID number. This infomration is stored in a simple space seperated file.

4.2.2 Accquired Data Storage

Accuiring fine grained data needed to keep track of power usage can eventually take up large amounts of storage space. This is not needed because fine grained data is usually only needed for events that happened recently. Trends become more important for data that spans large periods of time. This means that data can be agrigated into averages that span larger and larger periods of time the farther back in time one wishes to look. This is why RRDTool stores data that comes back from the PowerMole. RRDTool automatically agrigates data in this fashion. Data is stored inside individual Round Robin Databases (RRD) indexed by the system ID number.

4.3 Display and Software User Interface

4.3.1 Tabular Display

For each device, the user may select to display the gathered information in a table with plain text. Both current and power are shown to the user. The user may then select the time period that the PowerBurrow should report on as well as the granularity of the data. It should be noted that because old data is agrigated, the farther back in time one wishes to see, the level of detail decreases. From this point the user may also request the latest data (poll the device), change the description of the device as well as other device management capabilities. This screen is shown in figure 4.

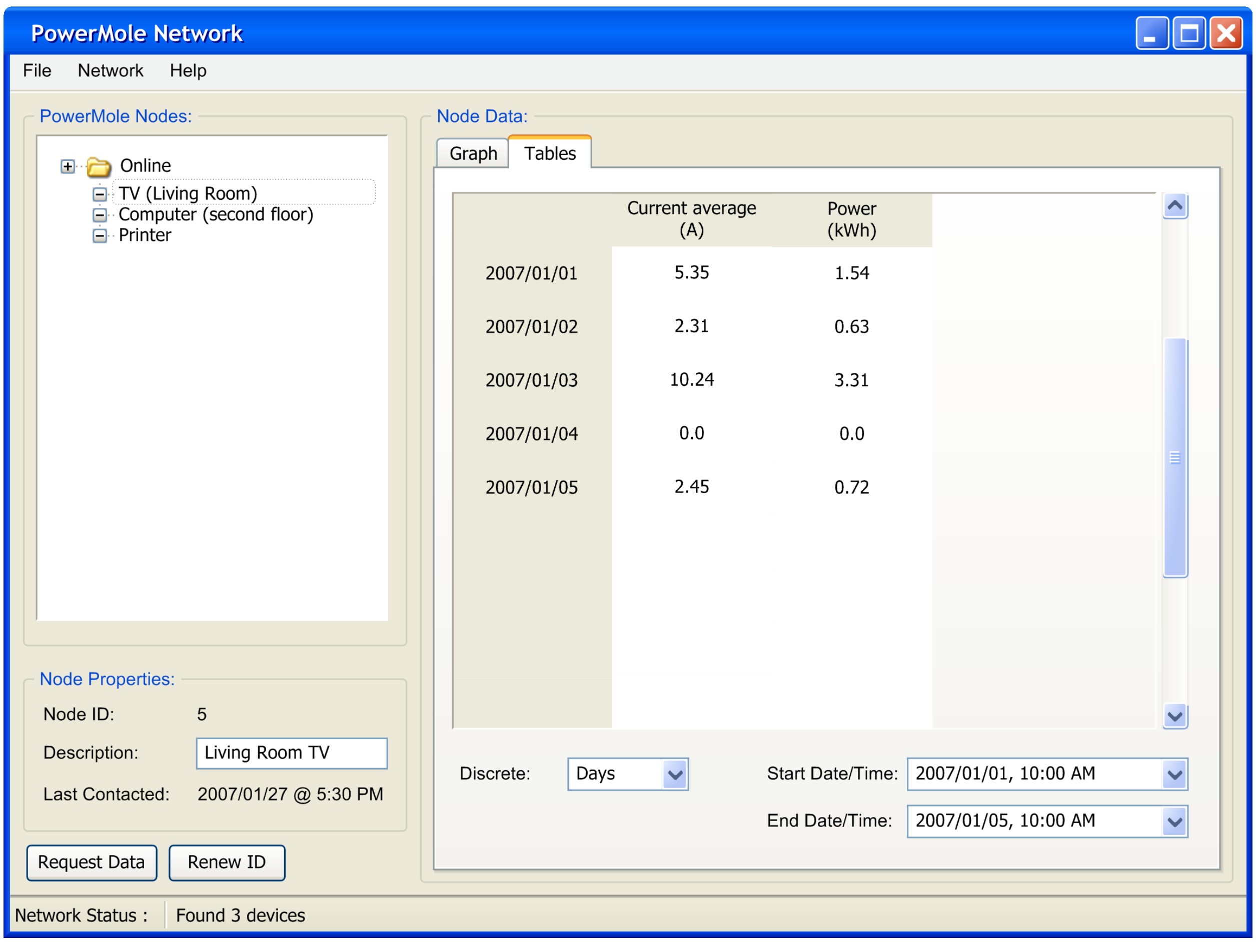


Figure 4: Software Interface - Table

4.3.2 Graphical Disply

For each device the user may select to display the gathered information in a graph. The user may select to display either current or power. Both current and power are shown to the user. The user may then select the time period that the PowerBurrow should report on as well as the granularity of the data. It should be noted that because old data is agrigated, the farther back in time one wishes to see, the level of detail decreases. From this point the user may also request the latest data (poll the device), change the description of the device as well as other device management capabilities. This screen is shown in figure 5.

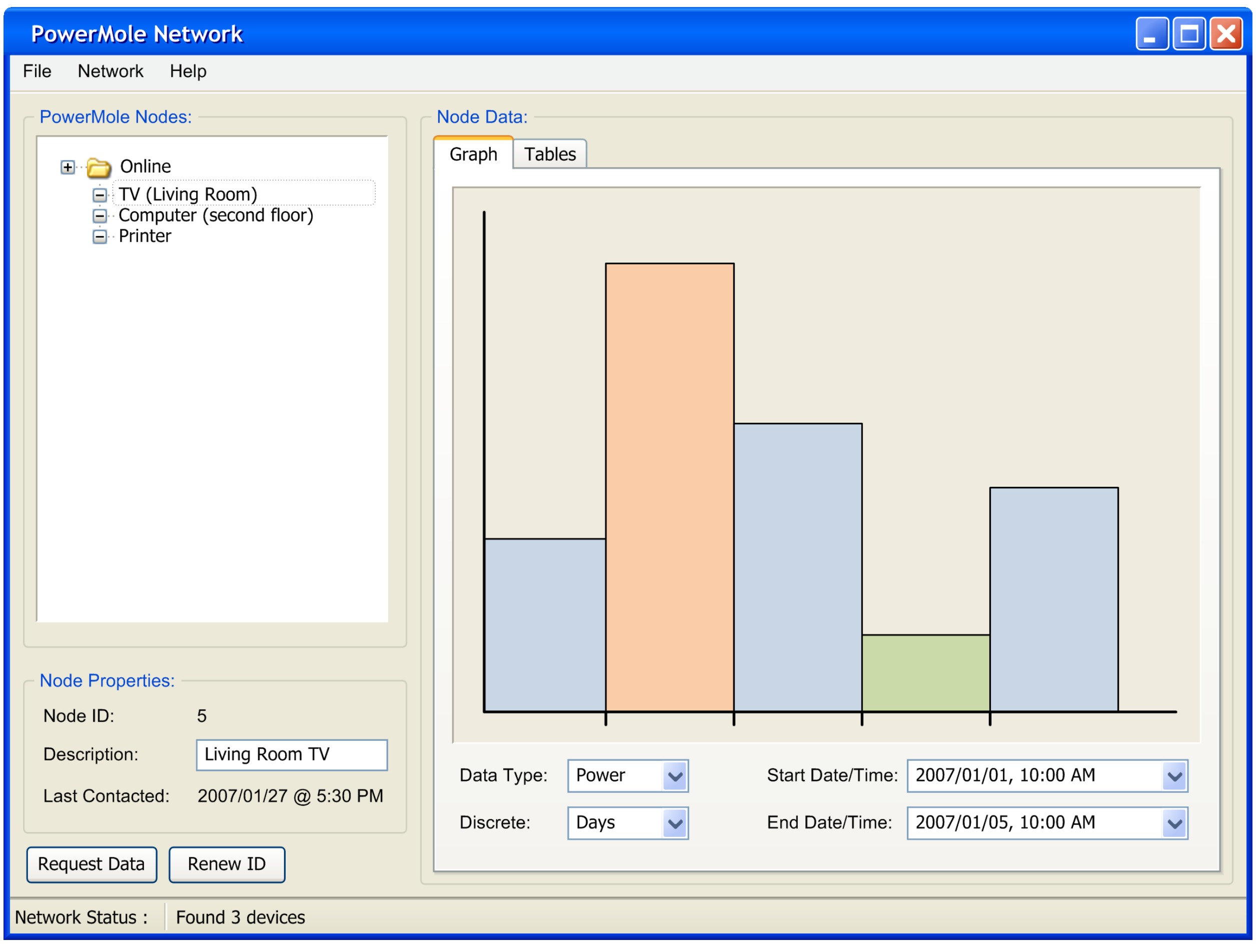
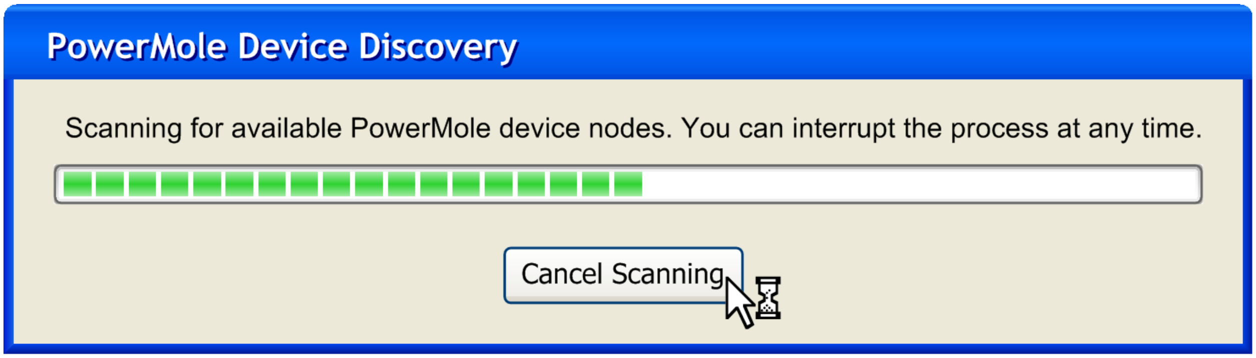


Figure 5: Software Interface - Graph

4.3.3 System Management

From the interface the user will be able to turn the network on and off, scan for new devices and export raw data. This is shown in figure 6



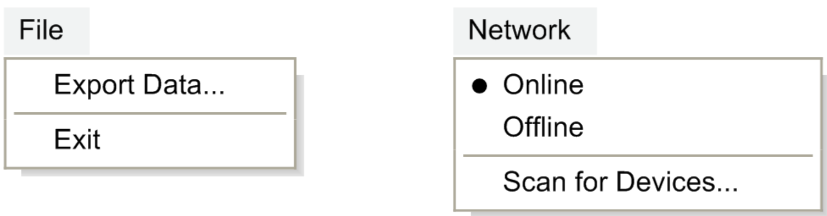


Figure 6: Software Interface - System Management

5 Deviation From Original Design Specification

This section describes the differences between the prototype PowerMole and the design specification [5].

5.1 PowerMole

5.2 PowerBurrow

5.2.1 Power Consumption Data

Data is automatically sent from the PowerMole when its internal data storage fills up. [5] called for the PowerBurrow to poll the PowerMole periodically. At first only the averages of the aggregate data was going to be stored. With very little work, the RRD was expanded to include minimum and maximum values of the aggregated data. This data is simple to add and may prove extremely useful.

5.2.2 Device Information

Originally, we planned to use DataDraw as the backend for storing the device information. After taking a close look at DataDraw, it became apparent that it was overly complicated for our needs. Instead a simple interface that would write a space seperated file was made. Originally [5] called for a much more detailed list of parameters to be stored. It was decided that this was unnecessary for our project, so a smaller amount of information about the device is stored.

6 Future Plans and Recommendations

6.1 Additional Features

As the PowerMole is currently only a proof-of-concept device, it only has a very minimal set of features. Expanding the cababilities, robustness, and minimizing the cost of components are needed before the PowerMole can be brought to market.

6.1.1 Data Gathering

On the PowerMole side, data gathering has two major limitations: limited space and volatile storage. The PowerMole currently holds less than five minutes of data. If there is a problem communicating with the PowerBurrow old data is overwritten. If the PowerMole looses power then all collected data is lost. Adding a memory card would help with both problems.

The accuracy of the Veris H722HC Current Transducer is listed as 2% of the maximum current, in this case, 50A. Because of this, low power devices will require several windings around the core to produce measurable current. While this is far from ideal, it was sufficient to demonstrate the proof of concept for the prototype design. For full scale production, a PCB transducer with greater accuracy and smaller size, such as the FHS 40-P/SP600 from LEM, would be ideal.

6.1.2 Wireless

Since we're using the standard ZigBee stack for wireless communciation, it automatically provides us with essential networking utilities such as authentication schemes, security, join/leave notificiations and others. Because of that, the improvement of wireless connectivity can be focused on the higher data management level. Since presently we have only one functional PowerMole device, it proved impossible to test the basic multi-device functionality. In future project iterations, it is necessary to produce at least 5 to 10 PowerMole devices and concentrate on delivering a fault-tolerant multi-device data delivery from PowerMole nodes to PowerBurrow.

Additionally, the current ZigBee stack implementation proved to be an unreliable one. Therefore, one of the top priorities is to investigate the alternative off-the-shelf ZigBee solutions to minimize the development time and get a more complete ZigBee stack, which is backed up by stronger technical support team.

6.1.3 User Interface

The addition of a web based interface would make our application more portable and would allow users to check how much power is being consumed while away from the PowerBurrow. This would also allow teams to easily share current results over large distances.

6.1.4 Application Programming Interface

The addition of an application programming interface (API) would allow users to easily plug their own tools and applications to the PowerBurrow. This would allow customers to integrate our software with theirs and greatly enhance the appeal of our product.

7 Budgetary and Scheduling Comparisons

7.1 Budget

Table 1 compares the proposed budget in our project proposal [3] to our actual spending on the prototype. In our estimated budget we did not consider things such as shipping,

Table 1: Comparison of Actual and Esitmated Costs

Parts	Estimated Cost	Actual Cost
Communication hardware	$150	$214.94
Power measuring circuit	$40	$88.64
Replacement parts	$60	$41.00
Miscellaneous parts	$60	$40.00
Total	$310	$384.58

duties, taxes and the cost of exchanging currencies. These are the source of the discrepency between our estimated and actual cost of communication hardware. We also did not consider that we had to buy additional hardware such as antennas. Initially we expected to make our own power measuring curcuit, but we ended up purchasing the Veris H722HC Current Transducer to do this for us. The difficulties encountered with the Integration ZigBee modules and dongle brought the decision to consider changing over to Xbee. To speed up the transistion a Xbee module was purchased. Eventually enough support came from Integration to get the Zigbee components working properly, so the Xbee module was a wasted purchase. Not including the Xbee module our cost overrun was only abot \$30. Even though this is a cost overrun of 10%, a budget as small as ours a small mistake accounts for a significant portion of the total cost.

7.2 Schedule

Figure 7 is the proposed schedule, taken from [3].

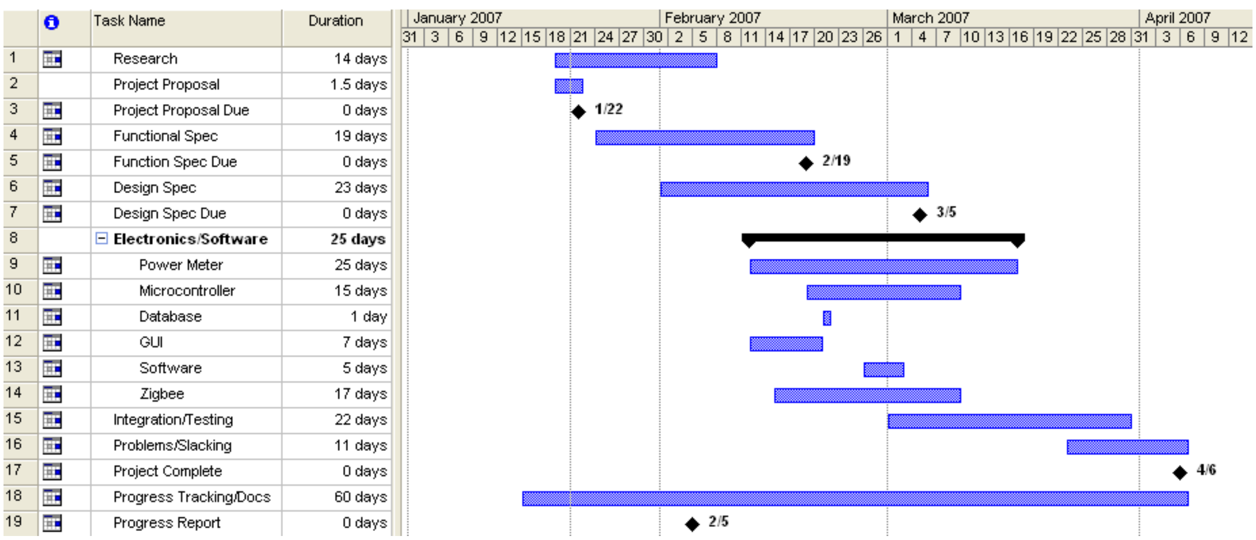


Figure 7: Proposed Gantt and milestone chart

The changes occurred mostly in the *Electronics and Software* section. Since we had issues with getting hardware working, the ZigBee, database and GUI development continued up until the presentation date. However, it can be also attributed to the *Integration* stage. Either way, the deviation from the planned schedule did not impact the result and the product was presented at the initially set date.

8 Interpersonal and Technical Experiences

The following section contains individually written descriptions of each members contribution to the project, what was learned, and what each team member would do differently if they were to undertake a similar project again. This section also includes a description of group dynamics within our team.

8.1 Dan Ralph

I was primarily responsible for interfacing the PC with the ZigBee dongle, achieving device to PC communication over ZigBee, and programming the user interface. On a technical level, I learned a lot about wireless communication over ZigBee, GUI building with GTK, and designing with microcontrollers. On a less technical level, I learned the importance of choosing vendors wisely. Our Zigbee devices were riddled with problems and were easily the most troublesome elements of the design. It took us nearly a week to get basic communication working while colleagues, who chose a different ZigBee solution, had much less difficulty. Features that were explicitly mentiond in the documentation, namely the power saving features, turned out to be incomplete. Documents regarding the API were filled with errors and often lacking adequete detail. Overall, we could have saved ourselves a lot of time and pain by choosing a better vendor for our ZigBee hardware.

8.2 Dimitri Tcaciuc

Since I was reponsible for the entire hardware development, my learning experience was mostly focused in that area. The most important lesson I got out of last four months is that rushed planning will always haunt you for the rest of the project. Because hardware development naturally has a high turn-around time, especially when constrained by limited resources, it is important to invest an extra time into researching alternative designs, components and suppliers. We learned a hard lesson that the specification and feature sheets aren't to be trusted lightly, especially when it comes to complex components which involve microprocessing unit with firmware. It is highly important to have a back-up plan so that if current design hits a brick wall because of one badly designed off-the-shelf component it would be possible to switch over to plan B in minimal time and continue the development.

8.3 Adam Leszczynski

As Cheif Operations Officer I learned a great deal. I was responsible for ordering, paying for and recieving the majority of our components. This helped me gain a better understanding of cross border purchasing and to better make budgetary forcasts. This course was also the first time that I ordered anything online. I spend a great deal of time working with Open Source Software (OOS) on Linux at my job. During this course a variety of the tools that I have used were an option for several components. Utilizing these pieces of software would save us development and testing time. I learned about the issues in porting some of these tools to Windows and I learned how to solve some of them. Most of my software projects are small self contained projects. This course helped me expand my experience in integrating and interfacing with componets made by other people. I have not touched C/C++ in a long time, and it was nice to refresh my knowledge of this programming language. Throughout this course I learned a great deal about programming and debugging mircocontrollers. I also learned the value of starting documents early. Simply filling out sections like Introductions, System Overviews and making stubs for later sections and having a template for format gets the document off to a good start. Details can be filled in as they come along, and it makes it easier to focus on the details on the document rather then presentation. This also makes it easy to determine the progress of the document and knowing how much time is needed to complete it.

8.4 Description of Group Dynamics

The team of Intueor Systems worked very well together. There were no major interpersonal conflicts throughout the course of the project. People worked on their sections and when help was needed the other memebers of the team provided it if they could. When it came time to integrate components there was no bickering over who had to change their components to make things fit together and work. Whoever had to do less work to make the systems compatible was the one that made the changes.

9 Conclusion

The last four months our team spent working on the PowerMole project were quite difficult. Hardware and software design, component selection, interactions with component suppliers, time management and task division were all the project elements that we had a full exposure to on a level we haven't faced before during our university carreer. It is extremely satisfying to realize that we have gone through this experience with success and have not only completed a project course, but have also came up with a working prototype of an innovative and truly useful product that can also be commercialized in the future. We would like to thank the ESSEF fund for helping with the necessary finances for our project and ENSC304/440 professors and TA's for giving the much needed guidance and support.

References

[1] BC Hydro - Quick Facts. Accessed Jan 20, 2007.
http://www.bchydro.com/info/reports/reports921.html

[2] Strategies for Engineering Communication. Stevenson and Whitmore. © 2002.

[3] Proposal for a Distributed Power Monitoring System. Intueor Systems. © 2007.

[4] Functional Specification for a Distributed Power Monitoring System. Intueor Systems. © 2007.

[5] Design Specification for a Distributed Power Monitoring System. Intueor Systems. © 2007.

[6] ZigBee Specification v1.0. Accessed March 4th, 2007. *http://www.zigbee.org*

[7] IA DAMD1 2400 datasheet. Accessed March 4th, 2007.
http://www.integration.com/docs/IA_OEM-DSMD1_2400.pdf