CPSC 483 – Computer System Design

Coff-e-mail Proposal

February 9, 2004

Don McGee
Eric Peden
Payton Quackenbush
Zack Roman

Table of Contents

Introduction

The coffee machine in the Computer Science Department lounge emits the very popular caffeine beverage consumed by a growing number of faculty, staff, and students in the department. The machine is shared by so many people that it is currently inconvenient to get coffee – no one knows whether coffee is left in the machine, how old the coffee is, or when someone has just brewed a fresh pot. Also, since it is frequented so often, it would be convenient to generate usage statistics of the machine. These statistics could also allow for justification of a secondary or more powerful coffee machine.

Enter *Coff-e-mail*. The Coff-e-mail project sets out to bring internet notification and web-based monitoring of the coffee machine in the lounge of the Computer Science Department. Coff-e-mail is an embedded system which will be non-intrusively retro-fitted to work with the current coffee machine in the Computer Science Department lounge (it will be designed to interface with *any* coffee machine). It includes sensors to detect when coffee brewing starts and the amount of coffee left in the machine (coffee level), along with a camera to take pictures when people fill their coffee mugs with fresh coffee. Coff-e-mail delivers a full-blown web server which can be accessed off the local intranet, which displays coffee statistics (pots of coffee brewed per day, last brew time, cups of coffee per brew, etc.) and pictures of people filling their coffee mugs. Client software will be available to poll Coff-e-mail and notify the user that a fresh pot of coffee is brewing or that coffee is running out and a fresh pot should be brewed.

Project Objectives

- Web cam interfaced to take pictures and store them in non-volatile memory
- Sensors to detect coffee brewing and coffee levels
- Web server accessible over the intranet which displays coffee status, coffee statistics, and web cam pictures

- Notification software running on Windows which will notify the user that coffee brewing has started or that the coffee level is low and a fresh pot should be brewed

The educational objectives for Coff-e-mail include embedded system design, network communications, web-based programming, software engineering, and ubiquitous computing.

Solution

Web-enabled devices have typically been thought of as "hobbyist" projects, and as a result most of the technical literature deals with developing new protocols and hardware for "smart appliances" and data acquisition devices (DAQs) rather than hooking web servers up to existing devices. [Che02] and [Sch01] deal with smart appliances; both articles discuss communication protocol issues. Because of our need for a web server, we chose instead to use TCP/IP for communication rather than a custom protocol, but [Sch01] provided a number of useful guidelines for building devices that stay out of the users' way. [Spa99] discusses an autonomous data-collection device that must deal with event-logging issues we also expect to encounter, such as timestamping and on-the-fly summaries of collected data.

[Oul01] details the development of a web server-on-a-chip. The product, Webchip, was one we were not aware of; after reading this paper and looking at the chip's documentation, it looks promising. [Yea03] discusses the development of a power-monitoring device. The device captures a variety of data points (temperature, voltage, current) and logs them on-board. Web pages are created by a CGI which runs on a dedicated web server. Again, this is not the approach we decided to use, but sample Java code further demonstrates logging techniques. Also, although the paper does not say what hardware was used, analysis of one of the screenshots suggests that the TINI controller provided network services to the device. Finally, [Wit98] provides an in-depth analysis of an old (circa 1998) embedded web server with attached web cam. It provides dated but real-world example of how to interface with a camera and alternate methods for creating dynamic web pages.

A low-cost embedded system for Internet based power measurement
Yeary, M.; Sweeney, J.; Swan, B.; Culp, C.
International Journal of Information Technology & Decision Making, Dec 2003.

A client-server architecture for distributed measurement systems
Bertocco, M.; Ferraris, F.; Offelli, C.; Parvis, M.
Instrumentation and Measurement, IEEE Transactions on, Vol.47, Iss.5, Oct 1998
Pages:1143-1148
URL: http://ieeexplore.ieee.org/iel4/19/16119/00746572.pdf?isNumber=16119∏=STD&arnumber=746572&arNumber=746572&arSt=1143&ared=1148&arAuthor=Bertocco%2C+M.%3B+Ferraris%2C+F.%3B+Offelli%2C+C.%3B+Parvis%2C+M.

Building smart services for smart home
Chemishkian, S.
Networked Appliances, 2002. Gaithersburg. Proceedings. 2002 IEEE 4th International Workshop on, Vol., Iss., 2002
Pages:215-224

URL: http://ieeexplore.ieee.org/iel5/7710/21127/00980856.pdf?isNumber=21127∏=STD&arnumber=980856&arNumber=980856&arSt=215&ared=224&arAuthor=Chemishkian%2C+S.

How to build smart appliances?
Schmidt, A.; van Laerhoven, K.
Personal Communications, IEEE [see also IEEE Wireless Communications], Vol.8, Iss.4, Aug 2001
Pages:66-71
URL: http://ieeexplore.ieee.org/iel5/98/20430/00944006.pdf?isNumber=20430∏=STD&arnumber=944006&arNumber=944006&arSt=66&ared=71&arAuthor=Schmidt%2C+A.%3B+van+Laerhoven%2C+K.

A flexible data logging device for wind potential measurements and statistical magnitudes
Sparis, B.D.A.
Electronics, Circuits and Systems, 1999. Proceedings of ICECS '99. The 6th IEEE International Conference on, Vol.3, Iss., 1999
Pages:1483-1486 vol.3
URL: http://ieeexplore.ieee.org/iel5/6565/17615/00814450.pdf?isNumber=17615∏=STD&arnumber=814450&arNumber=814450&arSt=1483&ared=1486+vol.3&arAuthor=Sparis%2C+B.D.A.

Providing network connectivity for small appliances: a functionally minimized embedded Web server
Riihijarvi, J.; Mahonen, P.; Saaranen, M.J.; Roivainen, J.; Soininen, J.-P.
Communications Magazine, IEEE, Vol.39, Iss.10, Oct 2001
Pages:74-79
URL: http://ieeexplore.ieee.org/iel5/35/20680/00956116.pdf?isNumber=20680∏=STD&arnumber=956116&arNumber=956116&arSt=74&ared=79&arAuthor=Riihijarvi%2C+J.%3B+Mahonen%2C+P.%3B+Saaranen%2C+M.J.%3B+Roivainen%2C+J.%3B+Soininen%2C+J.-P.

An Easy-To-Do Embedded Web Server [IC Online]
Witchey, N.
Internet Computing, IEEE, Vol.2, Iss.3, May/Jun 1998
Pages:100-100
URL: http://ieeexplore.ieee.org/iel4/4236/15040/00683808.pdf?isNumber=15040∏=STD&arnumber=683808&arNumber=683808&arSt=100&ared=100&arAuthor=Witchey%2C+N.

Design

Our design involves a microcontroller with on-board 10BaseT ethernet, a web cam, and sensors.

Microcontroller and Ethernet Options

We investigated a variety of microcontroller/ethernet solutions for the project, including both wired 10BaseT ethernet and wireless 802.11b ethernet. A table of the most promising solutions is shown in Table 1.

	TINI390 1 MB	SitePlayer	Iosoft ER22
RAM	1 MB SRAM (non-volatile)	512 B	1536 B
Flash Available	64 KB	32 KB	32 KB
Ethernet	10BaseT	10BaseT	802.11b
Programming Interface	Java 1.1.8, 8051 assembly	C, 8051 assembly	C, PIC assembly
Cost	$134.00	$99.00	$285.00
Web Site	http://www.ibutton.com/TINI/hardware	http://www.siteplayer.com/	http://www.iosoft.co.uk/wlan2.php

Table 1 - Microcontroller and Ethernet Options

Notice that each of these solutions utilizes ethernet access directly on-board. There are other options to interface a microcontroller and networking modules like the EDTP Packet Whacker[1], but these are less useful as they require developing all the networking protocols including ARP, IP, TCP, and HTTP for the microcontroller chosen. Thus, we chose the on-board ethernet route, which will let us focus more on things like the web cam.

Of the three options shown in Table 1, we find the TINI390 board the best option. It has a very large amount of non-volatile SRAM (1 MB), includes a Java programming interface, and is relatively inexpensive. This should allow us to develop rapidly and give plenty of room to store camera images and web content. The TINI board is a very popular board for use in ethernet embedded systems, and is well documented[2]. On the downside, it has a wired 10BaseT ethernet which requires plugging the system into an RJ-45 adapter with a CAT5 cable.

[1] Product and documentation available at http://www.edtp.com/
[2] The TINI Specification and Developer's Guide is available at http://www.ibutton.com/TINI/tinispec.pdf

The alternative option is the Iosoft ER22, which has an 802.11b adapter built-in. This gives it the ability to connect wirelessly to the local network and the internet. On the downside, it has a miniscule amount of RAM and has very few dedicated I/O pins available to interface with external devices. It would also need to be interfaced with some solid-state storage like Flash RAM to store the camera images.

Another alternative is the JStick, available at http://www.jstik.com/. It is a little more costly than the TINI ($299 vs. $134), but executes Java natively with its aJile processor, resulting in it running approximately 100 times faster than the TINI.

Camera Options

We have researched cameras in order to find the best solution for the Coff-e-mail project. From the research done, 4 cameras were chosen. Details about the four cameras are listed in the table below:

	CMUcam	C3188A	VGA00AIT1	GameBoy Camera/ M64282FP
Operation Voltage	5VDC	5VDC	?	5.0V
Operation Power	200mW(Active)	120mW(Active)	?	15mW
Video Output	Digital 8 bit	Digital 8/16 bit	8 bit (JPEG Encoding)	Analog
Lens	OV6620, CMOS image sensor	OV7620, CMOS image sensor	VGA size CMOS image sensor	Mitsubishi M64282FP CMOS Image Sensor
Color/B&W	Color	Color	Color	B&W
Resolution/Pixels	80x143	664x492	640x480 or 80x60	128x128
Vendor	seattlerobotics.com	kitsrus.com	usbdeveloper.com	Used Part
Price	$55	$85	$50	~$20

Table 2 - Camera Comparisons

The main considerations when looking at the cameras were the interface type/output type, the resolution, whether the camera was color or black and white, and the price of the camera. We decided that the camera needed a serial interface so that it could more easily be integrated into our system. Cameras with a digital output were preferred to avoid converting the signal with an analog to digital converter. The resolution of the cameras needed to be looked at so that we could determine how much space it would require to store the images and how high of quality the images would be. The lower resolution cameras would not require as much space to store images as higher resolution cameras, but the picture quality would suffer.

The camera that seems to be best suited for our project is the VGA00AIT1. The camera is fairly cheap and it has the capability to take low resolution images that will save space. The camera also has a built in JPEG encoding module. Converting the images from the camera to a JPEG format would help to save space and it would be easy to integrate with the web page/server. The main problem with this camera is the lack of documentation. The documentation provided on the website does give some control commands for the camera, but that is about it. It does not give more complicated diagrams such as timing diagrams.

The other camera that could be used is the CMUcam. The CMUcam has the benefit of being low resolution which would reduce memory requirements. It is also a digital camera so it would be easy to interface with. The documentation for this camera is also decent. The main drawbacks are the price and the output format. The camera costs $109 which is more expensive than other cameras. Also the output format would need to be converted to another type such as JPEG to be stored and integrated more easily.

Another option for cameras would be to go with a camera that is used in a GameBoy. The GameBoy camera would be fairly cheap and it takes very low resolution images that would not take much memory to store. The main problem with this route is it would take a lot of extra hardware to interface with the camera. The camera outputs in an analog format, so the output would have to be passed through a converter to get a digital signal.

The final alternative is to go with a C3188A camera. This camera has good documentation and it has a digital output interface, but the camera takes fairly high resolution images. The camera takes images with a resolution of 664x492. This would mean the images taken would require quite a bit of space to store, or they would have to be compressed quite a bit.

Lever

Best Option

Fit a shroud around the lever so that it may only be pulled forward (not backward). Mount a flex resistor to the front of the spigot. When connected to an A/D converter, this will enable us to measure not only that the lever has been pulled, but also how far it has been pulled. From this the amount of liquid flow can be roughly estimated. This method would probably be minimally intrusive, and require very little maintenance.

Price	$10 – 20
Intrusiveness	2
Modification level	2
Utility	4
Reliability	4

*** Options are ranked on a scale of 0-5, with 0 being the least of that attribute, and 5 being the most of that attribute.**

Button

Best option

Remove the top of the coffee maker to expose the electronics of the controls. Clip on two alligator clips to the leads for the brew button (may require removing a small bit of insulation). Run the wires from the clips to the control board, and replace the top. This option would be non-intrusive (completely invisible to the user), and can be done with minimal modification.

Price	< $5
Intrusiveness	0
Modification level	1
Utility	4
Reliability	5

Second Option

This method involves mounting two push-buttons on top of the real toggle switch. The buttons would be affixed by removable glue. This method would be very cheap, but probably require high maintenance to re-affix the buttons when they come off. In addition, the method may not be very reliable, as the user could hit the buttons multiple times, without flipping the toggle, or they could flip the toggle without hitting the push buttons.

Price	< $5
Intrusiveness	3
Modification level	2
Utility	3
Reliability	2

Third Option

This method is to build (with wood or scrap metal) a covering box with a toggle switch that would fit over the existing switch. The faux button would have rods extending down to the real button. Sensor leads would be attached to the faux button. Flipping the faux button would mechanically flip the real button, and the sensor leads would signal to the control board. The faux button assembly would be affixed with magnets or glue. This option has very high reliability, but is somewhat intrusive.

Price	$10 - 15
Intrusiveness	3
Modification level	2
Utility	4
Reliability	4

Water Level

Best Option

By mounting a series of photo resistors/transistors and LEDs on the sight gauge of the water tank, rough estimates of the water level can be detected (full, medium, low). An A/D converter would computer the readings from the photo-devices. This solution is very cheap, but intrusive, as a series of fixtures would be mounted to the front of the coffee maker.

Price	$20
Intrusiveness	3
Modification level	2
Utility	3
Reliability	3

Second Option

Water level can be measured by a capacitance probe. By inserting a wire into the coffee pot, the capacitance between the wire and water can be measured and used to calculate the water level. This method would be very exact and slightly intrusive, but would cost almost as much as the web cam. The wire would be Teflon-coated copper, which should not pose any safety issues in making the coffee unfit for human consumption.
See http://hawthorn.csse.monash.edu.au/~njh/electronics/watersensor/.

Price	$50
Intrusiveness	2
Modification level	2
Utility	4
Reliability	4

pH Sensor

Very little information can be found on an embedded pH sensor. Most pH sensors are part of manufactured sensor kits. Additionally, there are safety issues involved in selecting a pH sensor that would not contaminate the coffee. Due to these issues, a pH sensor is not being considered at this time.

Proposed Design

Our proposed design includes a microcontroller with on-board 10BaseT ethernet, the CMUcam, a wooden box-encapsulated button, a flex-resistor sensor for the spigot, and a series of LEDs/photo-transistors to measure coffee level. We based these choices on the feasibility analysis of each component as shown in the previous sections.

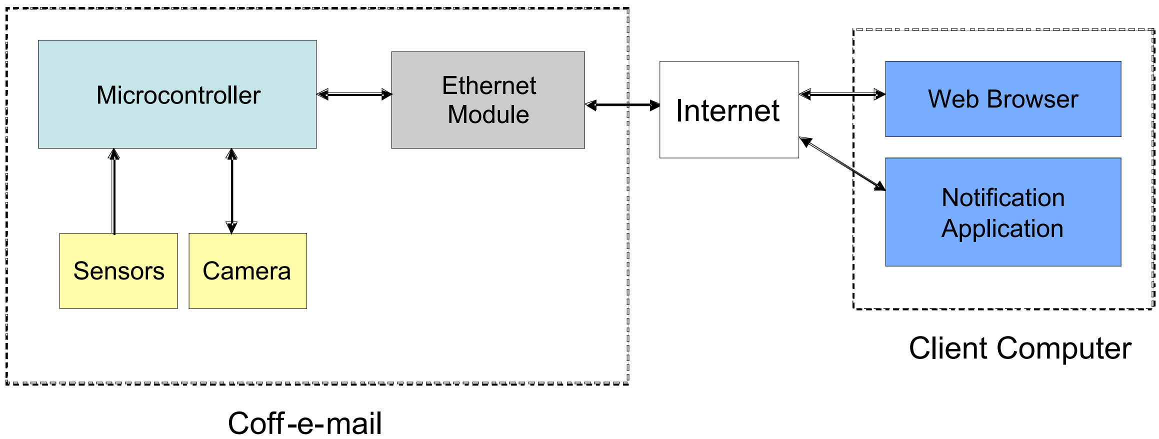


Figure 1 - Block Diagram

Item	Description	Cost
TINI390	Java-based microcontroller with on-board 10BaseT ethernet	$134.00
CMUcam	Web cam	$55.00
Spigot monitoring	Flex resistor	$10.00
Water level sensor	Infra-red LED/photo-transistors	$15.00
Brew button sensor	Push button assembly	$20.00
	Shipping	$30.00
	Total	$264.00

Design Validation

Testing web server – Start by writing a "*hello world*" application. We need to start simple and make sure the basics work, then we will start with the more complicated process of creating the web page with the statistics.

Testing for the camera – The testing of the camera will involve reading the raw data in off the camera into a file. This raw data will then be put into a file to see if the image can be displayed in a browser.

Testing the buttons – The buttons and sensors can be tested using a multimeter and reading the values off of the sensors.

Testing the environmental impact – We will install the sensors on the coffee machine to see what reaction we get from coffee drinkers. If the reaction we get is bad we might have to come up with a new plan.

Schedule of Tasks

Our schedule of tasks is shown below.

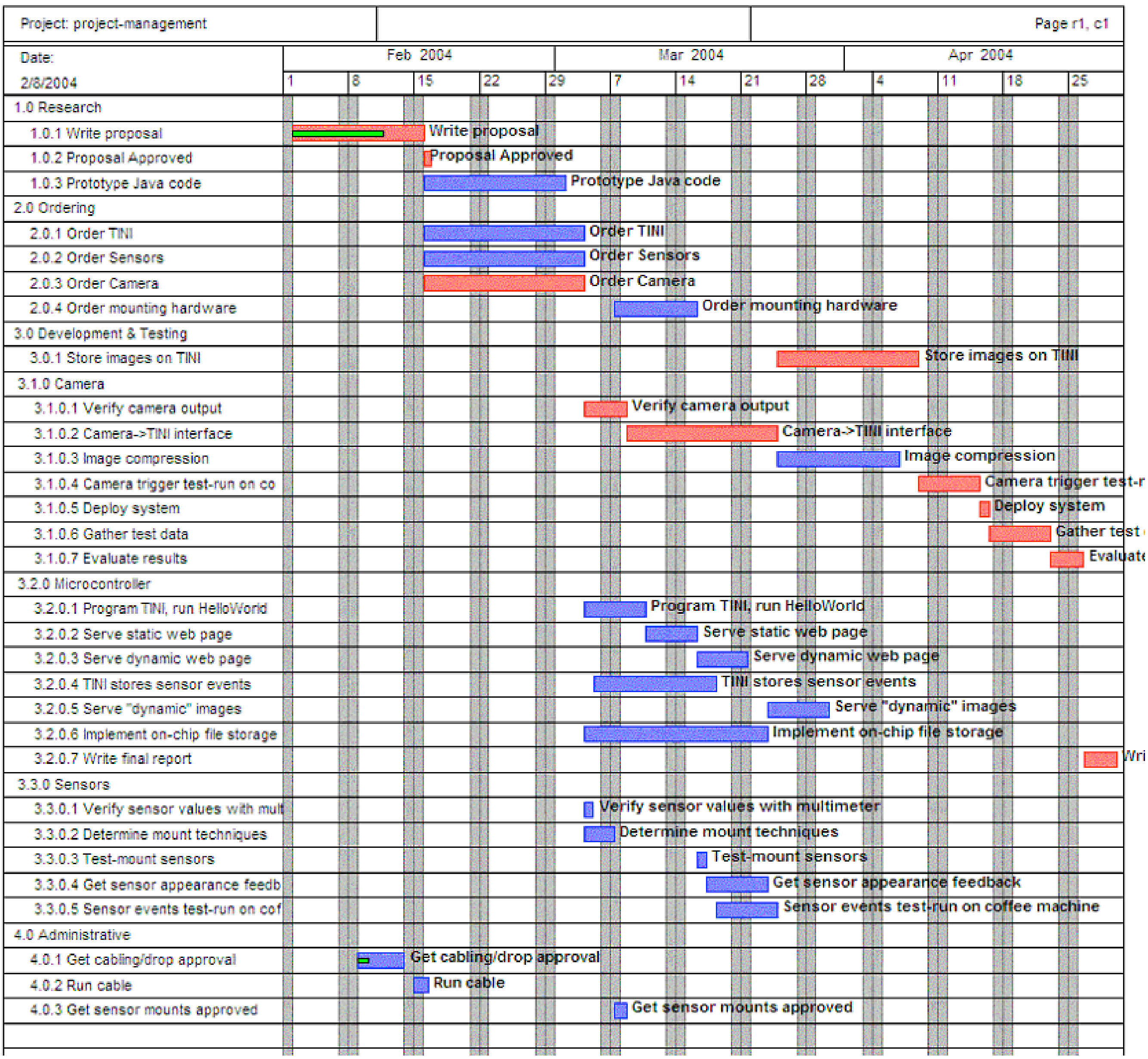
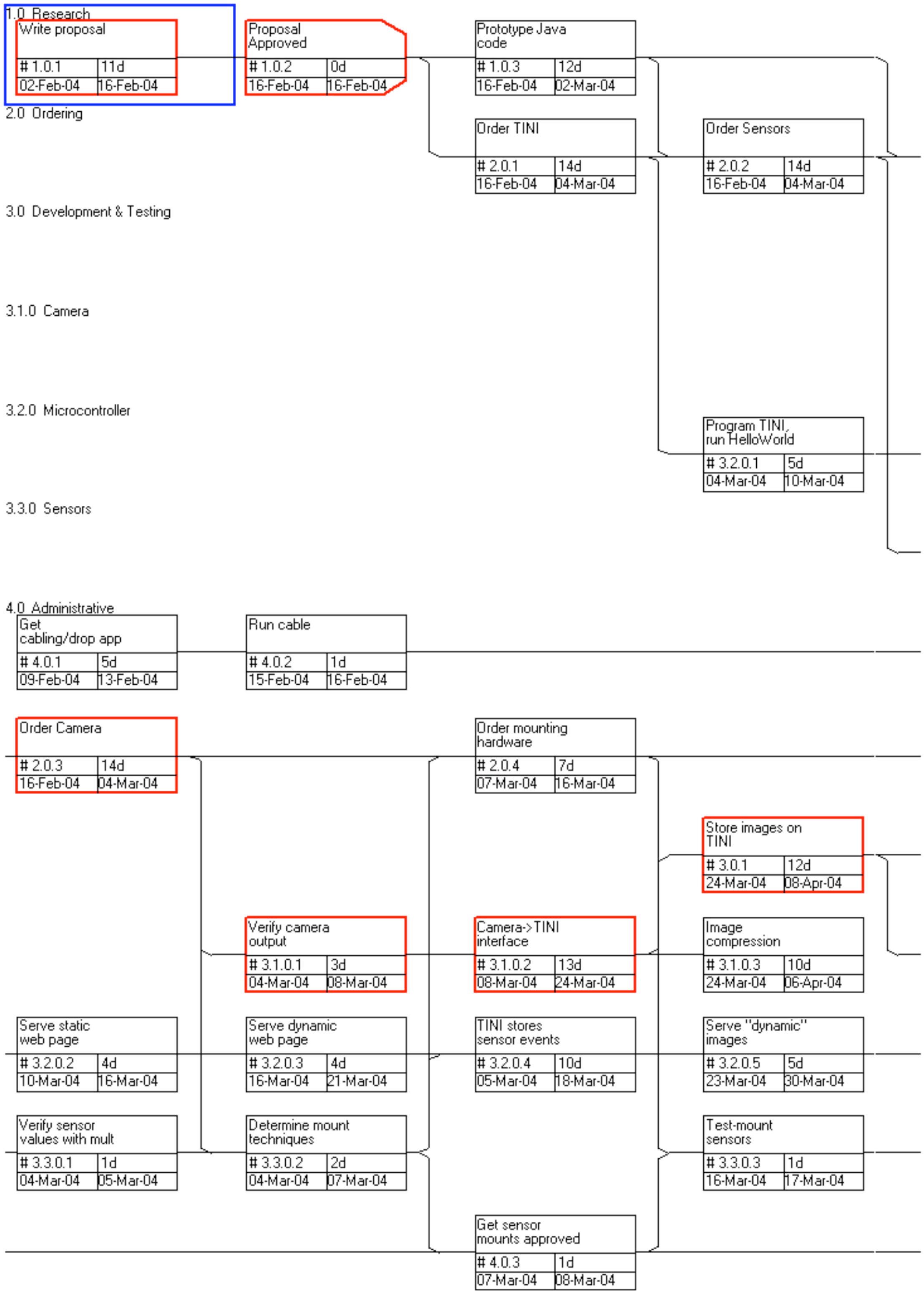
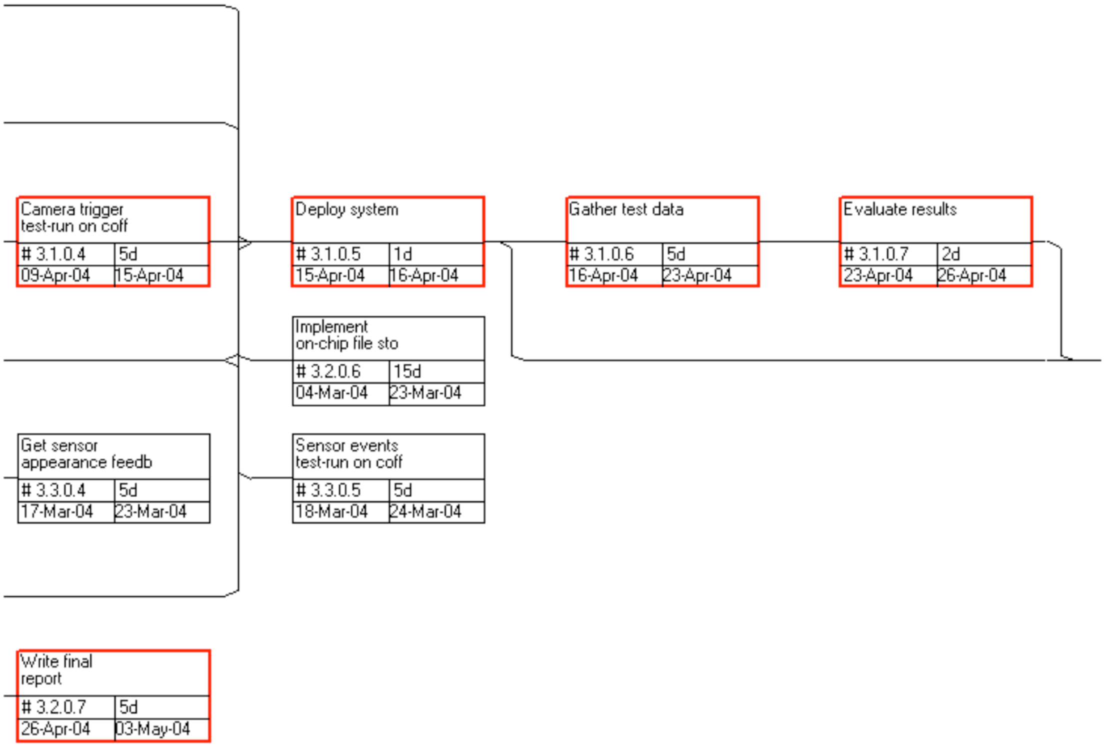


Table 3 - Gantt Chart (Timetable)

Pert Charts



1.0 Research
Write proposal
1.0.1 11d
02-Feb-04 16-Feb-04
Proposal Approved
1.0.2 0d
16-Feb-04 16-Feb-04
Prototype Java code
1.0.3 12d
16-Feb-04 02-Mar-04
2.0 Ordering
Order TINI
2.0.1 14d
16-Feb-04 04-Mar-04
Order Sensors
2.0.2 14d
16-Feb-04 04-Mar-04
3.0 Development & Testing
3.1.0 Camera
3.2.0 Microcontroller
Program TINI, run HelloWorld
3.2.0.1 5d
04-Mar-04 10-Mar-04
3.3.0 Sensors
4.0 Administrative
Get cabling/drop app
4.0.1 5d
09-Feb-04 13-Feb-04
Run cable
4.0.2 1d
15-Feb-04 16-Feb-04
Order Camera
2.0.3 14d
16-Feb-04 04-Mar-04
Order mounting hardware
2.0.4 7d
07-Mar-04 16-Mar-04
Store images on TINI
3.0.1 12d
24-Mar-04 08-Apr-04
Verify camera output
3.1.0.1 3d
04-Mar-04 08-Mar-04
Camera->TINI interface
3.1.0.2 13d
08-Mar-04 24-Mar-04
Image compression
3.1.0.3 10d
24-Mar-04 06-Apr-04
Serve static web page
3.2.0.2 4d
10-Mar-04 16-Mar-04
Serve dynamic web page
3.2.0.3 4d
16-Mar-04 21-Mar-04
TINI stores sensor events
3.2.0.4 10d
05-Mar-04 18-Mar-04
Serve "dynamic" images
3.2.0.5 5d
23-Mar-04 30-Mar-04
Verify sensor values with mult
3.3.0.1 1d
04-Mar-04 05-Mar-04
Determine mount techniques
3.3.0.2 2d
04-Mar-04 07-Mar-04
Test-mount sensors
3.3.0.3 1d
16-Mar-04 17-Mar-04
Get sensor mounts approved
4.0.3 1d
07-Mar-04 08-Mar-04



Project Management and Team Work

Milestones

Once the hardware has arrived, our project milestones are set up so that most development can take place in parallel. The TINI board is perhaps the most hotly contested resource, since at some point all components must interact with it directly. However, since components will belong to an individual (see *Source Code,* below) it will be easy to schedule testing sessions when it's time to actually run code on the TINI. Early on, each module can be tested with standard test equipment (multimeter, logic analyzer, PC) and we've built in some "wiggle room" at the end of the development cycle to allow for scheduling conflicts that might arise during final integration testing when all the components are brought together.

Source Code

Fortunately, we do not anticipate any major source code management issues with this project. Since Java will be used, code can be broken into classes that will belong exclusively to a single programmer. This eliminates the possibility of one person mangling another's code.

Team Assignments

The team has no designated leader; we in an anarcho-syndicalist commune. We take it in turns to act as a sort of executive office for the week. Team members are expected to ask for help if they need it, and to get their work done on time.

Meetings with the TA will take place every Wednesday after class. Team meetings will take place on Mondays during the scheduled class time and at other times as needed.

Don McGee	Camera interface, image storage
Eric Peden	File system, stats engine
Payton Quackenbush	Microcontroller programming, interfacing
Zack Roman	Sensor design, web design

Societal, Safety, and Environmental Analysis

Coff-e-mail has the potential to increase socialization of people in the Bright Building. By notifying users of fresh and available coffee, they can come together as a group, refill their coffee mugs, and discuss current issues, hobbies, or work. Overall, this has the potential to promote bonding, and further the wellness of its users. However, such benefits do not come with out a cost.

A primary concern is of course, safety. In operating any electrical device, there is always the hazard of electrical shock. Some of this hazard is innate in the coffee maker, and beyond our control. However, we must be conscious of the hazards our modifications will make. Therefore, we will endeavor to take utmost care in protecting exposed electronics and to minimize the potential for electrical shock. All exposed electronics will be protected or covered to accidental splashing from water (coffee).
The environmental impact we assume will be negligible. The coffee maker has been installed for a long enough time for everyone to have become accustomed to it. Likewise, as an appliance in an enclosed room, there is no additional affect on the outside environment.

As a final note, we would like to discuss health issues. Drinking coffee has numerous effects on one's body, most of which are *not* considered beneficial. By designing and building a device which essentially promotes the availability of coffee, we are in fact contributing to the lowered health of many of our users. How much a user drinks is beyond our control though, and we can only urge users to regulate their own coffee drinking.

Appendix A: Qualifications of Team Members

Each of our team members has taken CPSC 462, Microcomputer System Design, and has a solid foundation in Computer Engineering.

Don McGee has taken the CPSC462 class where some work was done with microcontrollers. Although the work done with microcontrollers was not very extensive, I think I learned enough that I will be able to fulfill the requirements of this project. I have also worked with C, C++ and Java quite a bit in the courses that I have taken. For Software Engineering, our team built a web solution using Java and Java Servlets.

Eric Peden is a senior computer engineering major. He has spent the past four years developing web pages, web applications, and distance learning applications for various TAMU departments. He has also done CGI development in Perl and Ruby, and is currently taking a networking class.

Payton Quackenbush has developed software for the past five years through summer internships and a Co-op with IBM. He created the "LCD information module" for his CPSC 462 group project, an embedded system which displayed information received off of a USB connection. He is experienced in C, C++, and Java, as well as web development including HTML, PHP, and JSP. He is currently writing a Nintendo Entertainment System emulator for Windows and Linux in his spare time.

Zack Roman has had experience with FPGA and microcontroller boards in Microcomputer Systems. He has also managed a Software Engineering project, and is currently employed by ATR, Inc. working on enterprise level web applications. He is concurrently taking a directed study involving communication between FPGA's and microcontroller boards.

Appendix B: Product Datasheets

- Microcontroller
  - TINI
    - Datasheet: http://pdfserv.maxim-ic.com/en/an/app195.pdf
    - Mechanical drawing: http://www.ibutton.com/TINI/hardware/mechrevB72.pdf
    - Dallas 80C52 microcontroller: http://pdfserv.maxim-ic.com/en/ds/DS80C390.pdf
  - SitePlayer
    - Datasheet: http://www.siteplayer.com/docs/001212/SitePlayer_SP1.pdf
    - Development Manual:
      http://www.siteplayer.com/docs/001212/SitePlayer_Development_Manual.pdf
    - Software Manual:
      http://www.siteplayer.com/docs/001212/SitePlayer_Software_Manual.pdf
  - IOSoft ER22
    - Datasheet: http://www.iosoft.co.uk/wlanfaq.php
- Camera
  - CMUcam: http://www-2.cs.cmu.edu/~cmucam/Downloads/CMUcamManual.pdf
  - C3188A:
    http://kitsrus.com/pdf/c3188a.pdf
    http://www.ovt.com/pdfs/ds_note.pdf
  - VGA00AIT1: http://www.usbdeveloper.com/Imagingkit/USBDeveloper.ImagingKit.html
  - GameBoy Camera:
    http://www.seattlerobotics.org/encoder/200205/downloads/M64282FP.pdf

Appendix C: Bibliography

- Microcontroller
  - The TINI Specification and Developer's Guide: http://www.ibutton.com/TINI/tinispec.pdf
  - TINI FAQ: http://www.ibutton.com/TINI/hardware/faq.html
- Camera
  - CMUcam: http://www-2.cs.cmu.edu/~cmucam/
  - GameBoy Camera
    http://home.earthlink.net/~apendragn/gbcam/
    http://www.smithstuff.net/image/Imagecap.htm
    http://www.seattlerobotics.org/encoder/200205/gbcam.html
- Sensors
  - Water Wave/Tide/Level Meter:
    http://hawthorn.csse.monash.edu.au/~njh/electronics/watersensor/