**reLive system**

*Critical Design Review*

reImagine Technologies

David Moreno

Mario Raushel

Terrence Sin

Caleb Wells

Department of Computer Science

Texas A&M University

March 3, 2008

Table of Contents

1 Introduction …………………………………………………………………………………………... 1

2 Proposed Design …………………………………………………………………………………………... 3

2.1 Updates to the proposal ……………………………………………………………………. ##

2.2 System Description ……………………………………………………………………. ##

2.3 Complete module-wise specifications ………………………………………………………… ##

3 Project Management ……………………………………………………………………………….. ##

3.1 Updated implementation schedule ………….…………………………………………….. ##

3.2 Updated validation and testing procedures …………………………………………….. ##

3.3 Updated division of labor and responsibilities …………………………………………….. ##

4 Preliminary results …………………………………………………………………………………………... ##

1. **Introduction**
   1. **Problem Background**

Diaries and logbooks have been around since the beginning of recorded history. Today diaries and logs have extended onto the internet through blogs and social networking websites. In order to create an accurate representation of the day’s events, one has to spend time writing in fine detail, which requires time they may not have to spend. Our system solves this problem by providing a complete photo log of daily events tagged to GPS coordinates, which is then viewable online through Google maps. This is accomplished by having the user wear a combination camera and GPS unit designed to take photographs at designated triggers: time, distance, or face detection. Images can then be transferred from the camera to a user’s computer and eventually the internet before being incorporated into their daily log.

* 1. **Needs Statement**

There is a need for users to autonomously create and explore daily photo life logs for both a fun and interesting experience.

* 1. **Goals and Objectives**

The goals for this project is to create a wearable camera system capable of recording GPS data that will automatically record the daily life of the user based on a variety of triggers including time, distance, and face detection.

Here is a list of objectives we have taken into consideration when designing our device:

* The prototype design should cost no more than $500.
* The system must be battery powered and capable of logging an entire day without needing to be recharged.
* The camera unit should not cause any harm to the user
* The camera should function well both inside and outdoors
* The GPS should function well in outdoor urban areas and have a decent failover for loss of signal indoors
* The design must be comfortable to wear and lightweight
* The accompanying software must be easy to use and understand
  1. **Design Constraints**

The economic constraints of our design are twofold. The initial prototype development has a total budget of $500. To be a viable commercial product, the final design must be both competitively priced and affordable to the consumer while maintaining a profit. In order to meet those specifications, the final consumer product should cost around $400.

As a wearable device, there are several physical constraints that must be met. First and foremost, the device must be lightweight and comfortable to wear for an entire day. As a wearable device, the system must run on batteries and be capable of powering the device for an entire day without being recharged. The placement and charging of the system’s batteries must also be easy to access and replaced as they will have to be replace regularly. In addition, access to the removable SD card must also be easy to access as users may want to synchronize pictures on a daily basis.

The design of the accompanying software must also be user friendly and easy to access all of the configuration settings of the camera. Anyone with basic computer experience should have no problem understanding how to use the program without much instruction. For the user to gain full use of the software however, the user must have an active Google account and password. This will enable the user to host their life log online where they can easily share images and accompanying maps to their friends and family.

The system must be usable both inside and outdoors, so the consumer product must be resilient to a variety of weather conditions.

* 1. **Validation and Testing Procedures**

There are two main components that need to be tested, the hardware and the software. To validate the hardware works we would have to test the GPS unit’s accuracy, the CMUCam3’s camera’s picture quality and responsiveness, and the microcontroller drivers’ logic correctness. To validate the software we must test first of all the logic correctness by running the software with various sample cases. We will test whether the program is reading and writing to the SD card correctly. We must make sure that all functions that interact with the user’s Google account are correct according to the Google API. To completely validate the software we need to test its ability to handle all the files without destroying any of them.

1. **Proposed Design**

**2.1 Updates to the proposal**

Our design has been completely changed from our initial proposal because after examining our initial proposal, the proposal was less a design than it was a research project. Our new design is now one of a wearable GPS camera (Cmucam3) that logs images throughout the day. The images will be mapped through a software application by their date, time and location. In effect, GPS data will have been incorporated with the video life log idea allowing users to see their pictures both chronologically and geographically.

Since the conception of the idea, there have been numerous changes to our initial design. Most of these changes have occurred in the implementation of the software application as it was difficult to do testing on the hardware design without the camera itself. For our application, we were initially looking into running a Perl script that would create the website’s html. This website would include all the Google maps features and functions we would want implemented.

However, after looking a little more into what resources we had, we found an API for Picasa web albums. Through this, we have implemented many of the features we had first desired through a C# application that interacts directly with a user’s Picasa Web Albums account. The application uploads pictures and retrieves the user’s album information to and from their account. One difficulty presented by using this API is in changing how the map is presented. More work will be done on trying to make this possible and allow the user different ways of viewing their albums and locations.

Some other changes include added configuration settings, features, and different ways of sorting albums and directories. On the hardware side of things, one important change has been made. Initially from the specifications, it seemed as if the GPS chip could be powered directly by the Cmucam3’s board. However, once we received the Cmucam3, we found the voltage between the pins we needed to connect to the board to be too high (> 3.3 V). Therefore, we will now be using a voltage regulator.

Because our design is completely different, a new budget will be presented below. The design consists of a programmable camera called the Cmucam3, a GPS chip, an antenna for the chip, and a SD card for storing the pictures taken by the camera.

**New Budget**

* Single Unit
  + Cmucam3 $ 239.00
  + Trimble Copernicus GPS Chip $ 74.95
  + Antenna $ 18.95
  + SD Memory Card ~ $ 15.00
  + Card Reader ~ $ 3.00
  + Enclosure ~ $ 10.00
  + 3.3 Volt Regulator ~ $ 1.00
  + Battery Pack ~ $ 5.00
  + Batteries ~ $ 10.00
* Total for 1 unit ~ $ 376.90

Accounting for extra costs that may come up, the total cost of our product will be around 400 dollars. This is within our goal of keeping the budget under 500 dollars. A large part of this cost comes from the purchase of the Cmucam3 itself and if bought in bulk, this price would most likely be much lower. For this reason, our product could probably be sold at the price of 400-500 dollars for a reasonable profit.

**2.2 System Description**

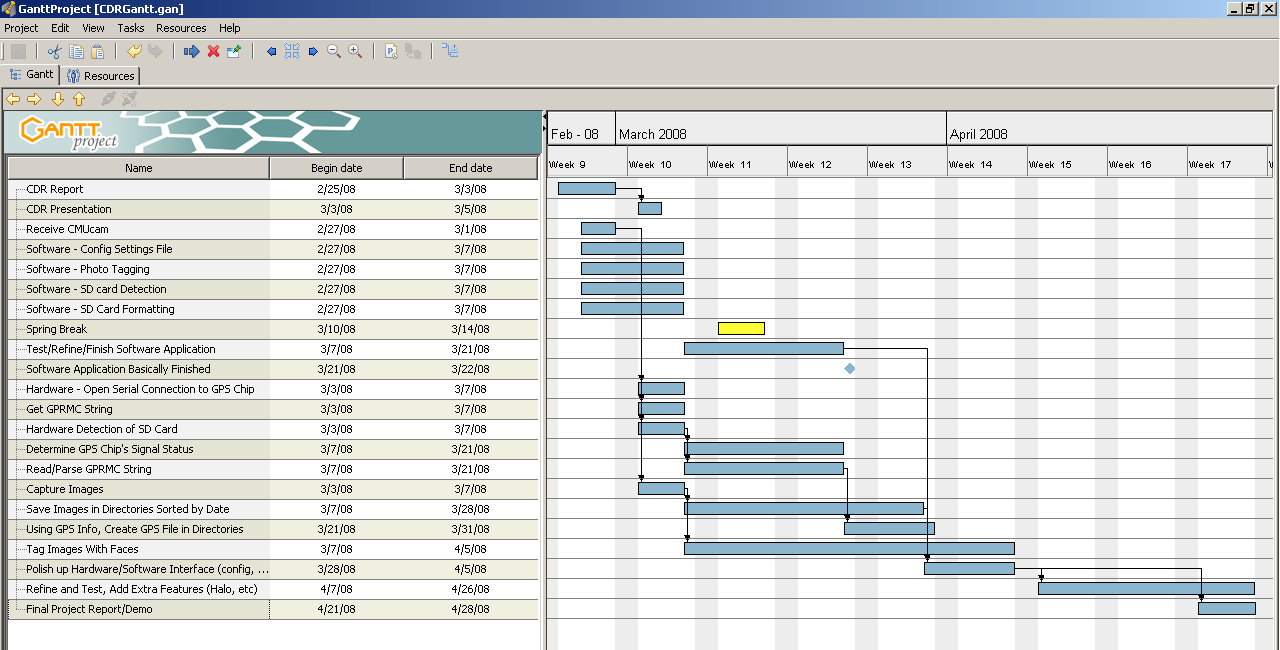
**2.3 Complete module-wise specifications**

1. **Project Management**

**3.1 Updated implementation schedule**

The updated schedule targets a completion date of all major features by the first week of April. In order to do this, there is a milestone of finishing major components of the software application by the third week of March, one week after spring break. Hardware components will take a little longer as we have just received the Cmucam3. Once the product is completed, we will consider the implementation of additional features such as the Halo feature and different map views. We will also need to test and polish the project, catching all errors that might occur.

The software part of the schedule includes the tagging of the photos, creating a configuration file for the Cmucam3, detecting a SD card, and formatting the SD card. During the hardware process, much of the software application will also be tested and refined. The hardware part of the process has just started and a lot of work still needs to be done. This includes opening a serial connection, getting the GPRMC string from the GPS chip, parsing it, capturing images, storing them in the SD card by date, and implementing the facial detection algorithm.

The final part of the schedule is the integration of hardware and software. This will be done through the configuration file and the SD card. The configuration file will be generated by the software, stored on the SD card, and detected by the Cmucam3 through the SD card. Both software and hardware sides will automatically detect the SD card and read off of it. The configuration file read by the hardware and the directories of images read by the software.

Shown below is the updated Gantt chart for the schedule.

**3.2 Updated validation and testing procedures**

To make the relive module a complete success every feature that is promised must work. In order to validate this product we will test its two main components, the hardware and software.

**3.2.1 Hardware**

**Copernicus GPS Test**

Testing the GPS unit will make sure that the location that it sends out is correct. We simply will take the GPS unit to various places, take a picture, and record where we are at. Once we upload the pictures to the PC, we can extract the GPS Data and plug it in to Google Maps if the location that it gives us is what we recorded then that will validate the correctness of the GPS Unit. This all relies on the fact that we were able to connect to the unit with the CMUCam3 and have received valid GPS strings, but more on that later.

**CMUCam3’s camera Test**

The camera is able to take pictures at two preset resolutions, high and low. Our product should take good quality pictures but it also need to be fast at it. To test this we will write a small program that takes pictures in both resolutions and keeps track of the time it takes to take and store each, this will validate the responsiveness of the system. We will also keep track of the pictures that have faces on them, to validate this feature we will have to take pictures of people’s faces in various environments and see whether or not the CMUCam3 is able to detect the faces. In doing that we would have also validated the quality of the pictures. Good quality pictures would make it easier on the algorithm to detect faces.

**Philips LPC2106 microcontroller Test**

Here we will mostly test the correctness of the logic of the drivers we write for this microcontroller. We will have to test that first of all the camera can take pictures when we want it to and store them on the SD card. We can validate that easily by pressing the button to take pictures, take out the SD card, load it into a PC, and see if there are image files. Second we want to store the GPS data with the picture, to validate that we have written a test function that will allow us to send a GPS string from the computer through serial. The microcontroller will then parse the string and send to the PC the GPS data through serial, if the string is invalid then it will send invalid. Finally we will test each of the triggers individually by providing the environment in which the trigger will be activated; if it takes the picture the system will be validated.

**3.2.1 Software**

**SD card management Test**

The SD card will hold the key to the interaction between the computer and the CMUCam3. To validate the correctness of the data that the SD card holds we will have to test our programs by saving test data and making sure the desired effect is produced from either end of the system. First we must make sure that the configuration file is correctly written and read into the CMUCam3, this will be done by outputting through serial every step of the microcontroller code. Finally we must test the CMUCam3’s ability to store images into the SD card, we will set up some triggers that will make the CMUCam3 take pictures, if we can see those pictures later on the PC then the SD card is validated.

**Google API Test**

To make sure that every function that handles the user’s Google account we will test them by inputting most the usual data that may come up through regular use. Our main concern is mishandling the user’s Google information. By following strict guidelines set through the API we should be able to validate our program is secure.

**File survival Test**

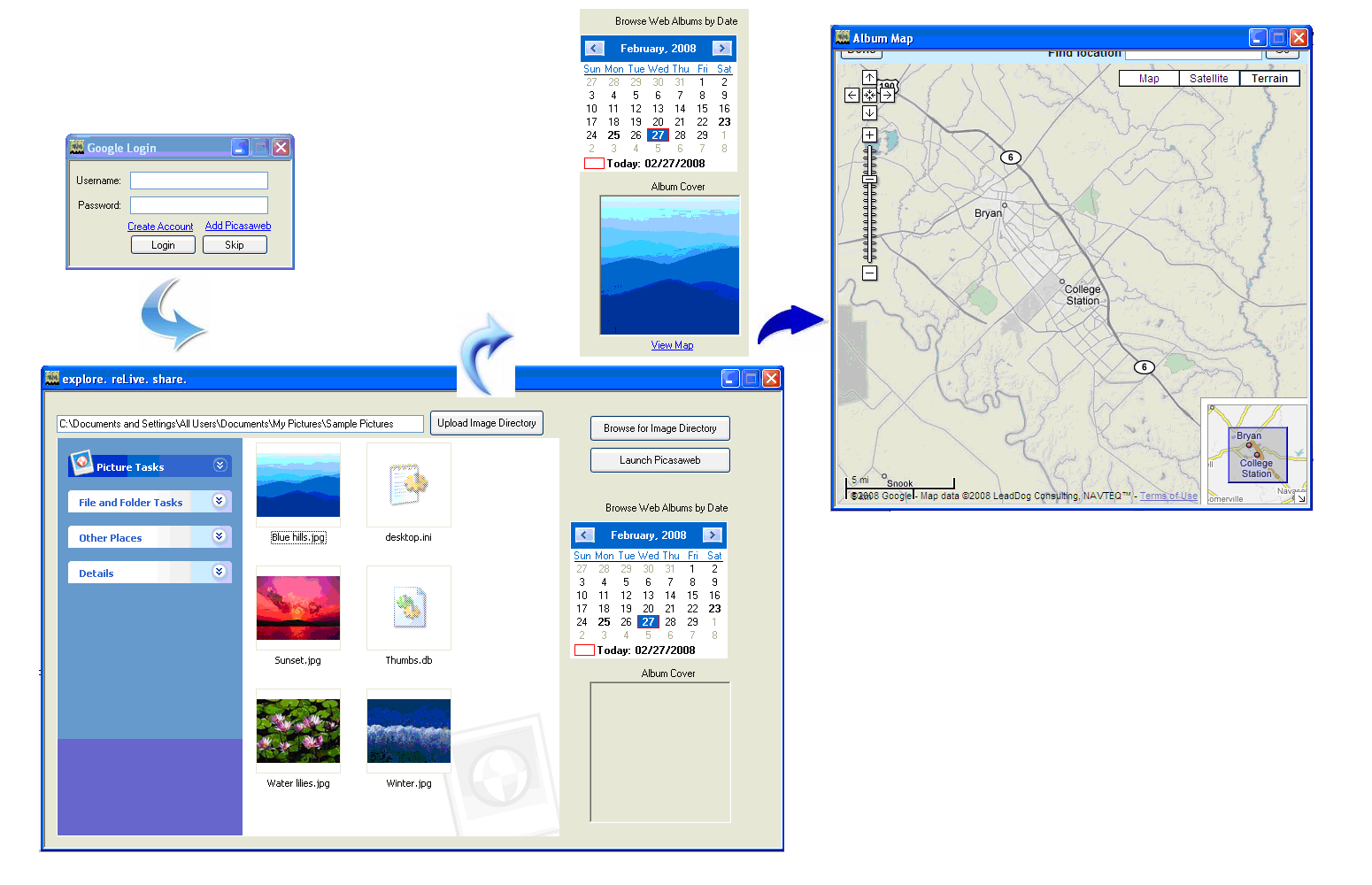
This is will be the easiest to validate. Essentially we need to make sure that our program will not destroy files in the computer, the SD card, or the Google account. To test this we will make sure our program interacts with each and every file that is related to our product in the above mentioned places. If the files are still intact after the end of the program then it is validated.

**3.3 Updated division of labor and responsibilities**

Terence Sin will work with Mario to complete the software application. This includes implementing the configuration settings, detecting the SD card/formatting it, and tagging the photos. Meanwhile, he will also be looking into features involving the face-detection capability of the camera and how those could be added to the project.

1. **Preliminary results**

Because we just received the Cmucam3, we have been unable to test and refine the code written for the hardware aspect of our design. However, this has also caused us to focus much of our efforts toward the development of the software application.

The application’s process is shown below

The first step window that comes up is the Google Login window. This is where the user logs in to his Picasa Web Albums account. There are also options of creating an account, adding Picasa web, and skipping the login stage. A user skipping the login stage may only desire to change configuration settings on his camera. However, he will not be able to access and browse his albums that have already been uploaded. The other two links merely link to the signup and Picasa web pages.

Once a user logs in, the main page of the application shows up. Currently, only the Picasa album interaction is implemented. The user will select a directory containing the images he wishes to upload. The default for this is the ‘my pictures/reLive’ directory or whatever directory our camera will be uploading into. The user can also browse his past albums by clicking on the bolded dates on the calendar. The album cover of those albums will be shown. Clicking view map will then popup the map of the photos with pictures showing up at their respective positions.