Adarza Technologies
Simon Fraser University
Burnaby, BC, V5A 1S6
Email: info@adarza.com

April 22, 2005

Lakshman One
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

RE: ENSC 440 Post-Mortem for the Bacteria Classification Assistant

Dear Mr. One,

The attached document, Post-Mortem for the Bacteria Classification Assistant, details the current status of the system, suggestions for future improvements, budget, and actual timeline that we followed. Comments by members of the team are included at the end of the document.

The final goal of the project was to have a proof of concept program that could identify different visual features of bacterial colonies and return a subset of database records matching those features.

Adarza Technologies consists of four senior engineering students: Edward Goh, Geeda Ip, Josna Rao, and Linda Wu. Between the four of us, we have a broad knowledge base from software engineering to molecular biology, making us very well suited to take on this project. Please feel free to contact us if you have any questions, comments, or concerns regarding this document. We can be reached by e-mail at *info@adarza.com* or by phone at 604-437-6599.

Sincerely,

Edward Goh

Edward Goh
President and CEO
Adarza Technologies

Enclosure: *Post-Mortem for the Bacteria Classification Assistant*

Post-Mortem for the Bacteria Classification Assistant

Project Team:
Edward Goh
Linda Wu
Josna Rao
Geeda Ip

Contact Information:
Edward Goh
info@adarza.com

Submitted to:
Lakshman One
Mike Sjoerdsma
Steve Whitmore
Scott Logie
Tony Ottaviani
Amir Masoud Niroumand

Date Issued:
April 22, 2005
Revision 4.0

Table of Contents

List of Figures

List of Tables

Abbreviations

BCCDC	British Columbia Centre for Disease Control
CC	Classification Criteria
OpenCV	Open Computer Vision Library

Glossary

ADO .NET	A set of data access API from Microsoft.
Agar	A gel usually made from livestock blood used to grow bacteria, fungus, etc. under lab conditions quickly and easily.
Bacterial Colony	A mass of millions of individual bacteria that has specific characteristics unique to that strain of bacteria.
Bacterial Strain	A specific species of bacteria. (i.e. Candaidas albicans)
Boundary Shape	The shape of the colony edge. Can be smooth, undulating or finger-like.
Colony Shape	The general shape of a bacterial colony. Can be circular or irregular.
Microscope Camera	A microscope with a digital camera attachment.

1 Introduction

The Bacteria Classification Assistant (BCA) concept is conceived under the guidance of Mr. David Chan and Dr. Swee Han Goh at the BC Centre for Disease Control (BCCDC). The aim of the BCA is to provide assistance to lab technicians when identifying unknown strains of bacterial colony cultures. Under normal circumstances, identification for an unusual strain can be time-consuming and expensive because of the various bio-chemical tests necessary to make an accurate identification (for more details regarding bio-chemical identification methods, please refer to our *Project Proposal* document). Our goal for the BCA is to streamline this identification process by providing the technician with a tool that eliminates a significant number of possible candidates based on visual criteria.

The BCA receives as its input, a digital image of bacterial colonies grown on agar medium, as well as three pieces of information provided by the user: a reference image of known size, a crop of a single bacterial colony, and a rectangle within the colony itself. Images of the bacteria are taken with a microscope camera and feeds directly into a computer where the BCA is running. The general setup of the system is shown in Figure 1. The BCA will then output to the user a list of possible candidates selected from a reference database plus information on each candidate strain. The four classification criteria (CC) used to generate a list of candidates are: size, colour, colony shape and boundary shape.

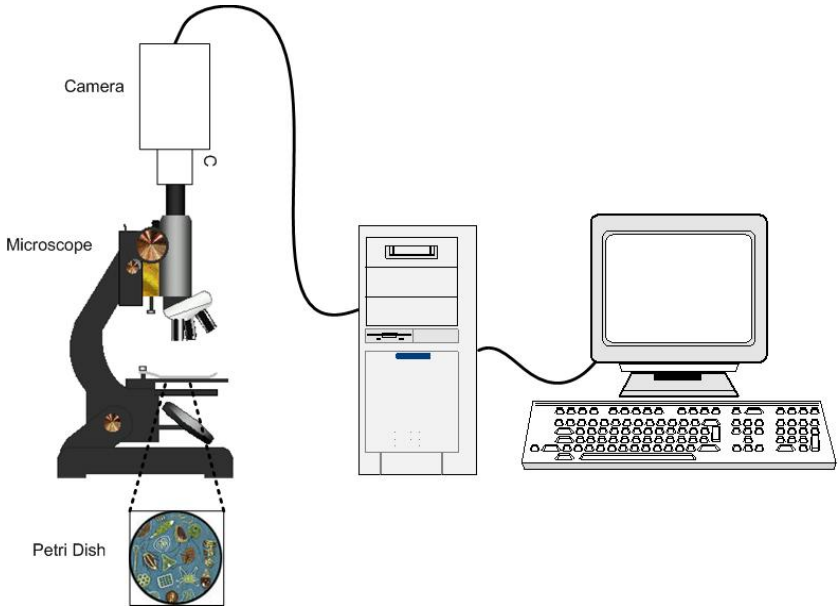

Figure 1: The Bacteria Classification Assistant System

To accomplish our aim, we used image-processing methods to determine a category output for each CC. A database stores the information on each classified strain and returns a list of possible candidates from the reference set. The GUI guides the user to analyze an unknown strain, make changes to the database and add additional information

into the database. A high-level block diagram of the BCA functional modules is available in Figure 2. The image-processing modules are classified into two categories: pre-processing and classification modules. Pre-processing modules provide parameters that the classification modules use to determine each CC.

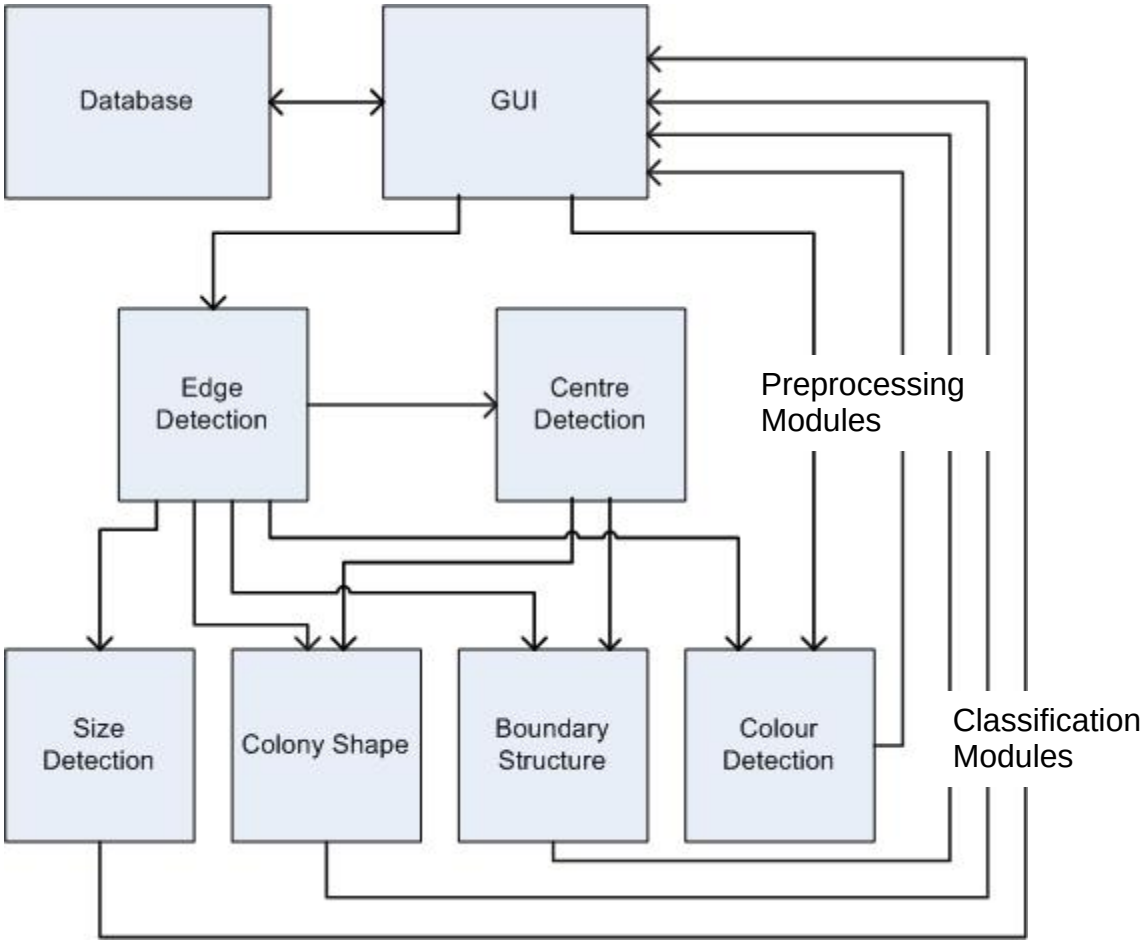

Figure 2: Functional Block Diagram of the BCA

Throughout the semester, Adarza Technologies has implemented each module of the BCA according to the goals outlined in our *Functional Specifications* document with varying degrees of success. Because each module is highly independent of other modules in its functions, we were able to work in parallel on several modules at the same time. Our parallel approach to implement the BCA allowed us to finish integrating our project and complete some testing within the time constraints.

2 Current System Status

The BCA is currently in its prototyping phase. The image-processing modules are able to output CC's for inputted images and the database is populated with 52 reference samples, along with their associated information. The GUI is capable of searching and displaying reference and sample matches. Overall, the BCA establishes proof-of-concept for use in the assisting bacterial colony identification under laboratory conditions.

2.1 Image-Processing Modules

There are six individual image-processing modules: two pre-processing and four classification modules. The two pre-processing modules are edge detection and centre detection. The four classification modules are size, colour, colony shape, and boundary shape. The following sections will detail the current status of each module and some of its possible improvements.

2.1.1 Pre-processing: Edge Detection

At the completion of the BCA, the edge module returns an image representing the colony in white and the background in black as specified in the *Design Specifications* document. The module returns an accurate edge if the input image is clear, without transparent or blurry edges, with an accuracy of 85%. If a blurry input image is entered, an unreliable edge image is returned. The edge module takes approximately 2 seconds to return an edge image. The module also requires a point coordinate on the inside of the colony in order to ascertain whether the contour is closed. The edge module could be improved so that the inside point is not needed, which would decrease chances of the edge changing depending on the inner point location. Since all of the classification criteria modules are dependent on the edge, overall reliability could be greatly increased by simply improving the accuracy of the edge module.

Since the edge sequence is dependent on the edge image, the sequence is returned with the same accuracy. The module uses OpenCV functions to identify the contours from the edge and links them in order. The points on the contours are identified and arranged in an OpenCV sequence object that can be used by the CC modules.

2.1.2 Pre-processing: Centre Detection

Finding the centre of a colony with a well-behaved edge image is consistent when using the centroid method outlined in our *Design Specifications* document. Test results for good samples in the database always return centre points that appear reasonable to the user. Low levels of noise are averaged out, but if a large section of the colony is lost

during edge detection, the centre finding module will be unable to find the actual centre. Total average processing time was one second for this module with some variation in time that is proportional to image dimensions.

Tests related to incorrect centre points have not been conducted at this point. While we do not want to use centre points that are not near the actual centre point for the image-processing, the effect on the boundary and colony shape histograms may yield information regarding the tolerance of the centre detection module.

2.1.3 Classification: Size Detection

The algorithm has been implemented as described in the *Functional Specifications* document and an additional field for a weak size category has been added to take into account the event of values close to a threshold. The thresholds are hard-coded into the module so in the future a more adaptive and statistical approach will be required. We have allowed for four size categories: tiny, small, medium, and large. By creating a new record in the size table of the database and adding more thresholds, additional categories can be implemented. However, redefining the categories would require repopulation of the size value for every entry in the database.

Sample pictures in the database that were taken on February 23rd did not have a corresponding picture of a calibration object. However, a subsequent batch of sample pictures taken on April 8th did have pictures of the calibration object constructed by our group. As a result, our current database has a reference object that was constructed using additional imaging software. The ratio between the reference and sample objects in both batches was consistent for many samples during testing. However, several samples that were present in both batches had significantly different ratios that placed the size in categories that were two steps apart, i.e. one strain was classified as large in the February batch, and small in the April batch. Depending on the dimensions of the image being processed, the module will run from between less than a second up to three seconds. After running the current calibration object through the edge detection module, we also noticed that the result was not smooth but had small finger-like protrusions. Currently, the size module is unreliable and needs to be retested with a new database with proper reference objects.

2.1.4 Classification: Colour Detection

According to our *Functional Specifications*, the goal of the colour detection module is to output a colour parameter from one of six categories (white, grey, yellow, blue, green, or red) along with a confidence percentage. In our current implementation, the colour module returns two parameters: primary and secondary colour from one of eight categories (white, grey, white-yellow-grey, yellow, blue, green, purple, or red). In cases where the module is highly confident of the colour, both the primary and secondary colour parameters will be the same. The changes were made to the output parameters in

order to better accommodate the actual colours of the sample images. From the samples we received from the BCCDC, we determined the need for an additional colour category: white-yellow-grey to take into consideration the colonies that are off-white in colour. The secondary colour parameter output replaces the confidence percentage because a second colour output is more useful for data filtering.

Currently, the module is about 90% accurate when identifying the colour of a given image provided the image has a good edge image. When the edge image is either much bigger or smaller than the actual colony, the module accuracy falls to about 50%. However, even though the module can accurately identify the colour of a given sample, the output colour parameter is not always useful due to the lighting conditions of the sample. The white, yellow, grey and white-yellow-grey categories are highly sensitive to the brightness and contrast of the image, and the output parameter can easily migrate between these four categories for the same bacterial strain depending on the sample data acquisition conditions. Even the other colour categories (red, blue, purple and green) can fade out into grey or white under extreme lighting conditions.

One important improvement that must be implemented is some way of rendering colour detection less dependent on the lighting conditions. There are several possible approaches to solving this problem. One possibility is to place some type of colour marker on the reference image (used primarily in size detection, currently) that will allow the BCA to derive lighting condition information and then normalize the samples. Another possibility is to average saturation or value of the entire image and then normalize colour selection parameters based on the averages. Both methods require additional experimental data and testing; however, either method will increase the reliability and accuracy of the colour detection module and the overall program.

2.1.5 Classification: Colony Shape Detection

The colony shape module is dependent on the centre point and edge sequence. Using the two input objects, the module constructs a histogram of the frequency of radial lengths. Analysis of the histograms revealed that circular colonies were characterized by high and narrow curves while non-circular colonies had a wider distribution. The implemented algorithm looks for the highest frequency and calculates the percentage of the total points that have a similar radial length. The module returns whether the colony is circular or not but additional analysis of the histograms may yield more categories that may be classified. If more categories are added, a second field to represent weak matches near threshold values may be required to improve reliability.

Currently, the colony shape output is unreliable. Some errors correspond to poor edge results but several samples still fail with a proper edge image. The histograms for the radial lengths have been constructed properly but the thresholds are still hard-coded and not dependent on any statistical algorithms yet. Further testing and implementation is recommended to correct the thresholds. Since fewer computations are required compared to the other modules, the colony shape can usually be calculated in less than a second.

2.1.6 Classification: Boundary Shape Detection

The boundary shape module returns a classifier for the edge feature of the colony. Edges are currently labelled as smooth, undulating, or finger-like. The boundary shape accepts a sequence of edge points from the edge module and calculates the angle between the normal to the radial vector and the tangent vector at each edge point. A histogram is then created from the edge angles. Currently a number of set thresholds are used to evaluate the histogram. The accuracy of the module is approximately 94% if the input edge is reliable (i.e., sharp edge on the original colony image), and the processing time to return a classifier is 1 second. For reliability, two classifiers are returned in case the edge is erratic, indicating a blurry input image.

The key future development for boundary shape would be to remove the use of set thresholds for analyzing the histogram. One method of accomplishing this would be by having standard histograms of smooth, undulating, and finger-like colonies that the test image's histogram would be compared to. These standard histograms would be directly stored in the program. Another method of removing set thresholds would be to make the thresholds adaptive based on a series of initial sample images. The user would classify what type of edges these sample images have, and the program would note key features in the histograms of these images. Based on what the user classified the images as, the program would adjust the internal thresholds. Another area for improvement is in the overall shape of the histogram. The histogram for the boundary shape module currently evaluates the edge angle as between 0° and 90° and does not separate negative angles from positive angles. Since this may cause a loss in data if the colony is not reasonably symmetric, a future development would be to correctly evaluate negative angles. Once this is done, statistical methods such as obtaining the standard deviation can be used to evaluate the histogram.

2.2 Database

The BCA has a fully relational database, consisting of eight data tables. The BCA can currently read, update, add and delete records successfully to the database. The actual database is stored as a Microsoft Access .mdb file, and its contents are accessed in code via ADO .NET. The database is expandable, allowing further classifiers and classification criteria to be added. The image files are stored in the \data folder, and image records in the database point to these files on disc. In the future, the image files may be stored as BLOB (binary large object) fields in the database, eliminating the need for the \data folder. However, doing so might reduce the speed of the database, and further investigation needs to be conducted.

Our current data access model is based on a disconnected client set, meaning that the BCA program does not have to maintain a constant connection with the database. However, this method requires that the database be loaded into memory beforehand, thus

requiring a protracted period of time for the program to load. In the future, we will want to investigate different modes of database access to improve load time.

The database is currently stored as a Microsoft Access .mdb file, which requires BCA users to have Microsoft Access installed on their computer. A future possibility is to move the database away from Microsoft Access, perhaps using XML, so that computers without Microsoft Access installed will still be able to run the BCA.

A desirable future feature for the database is reporting capabilities. Users will be able to create reports based on CC classifiers (for example, a report listing all bacteria in the database that are small, finger-like, and yellow).

Another possible future feature is access control. Users will need to log in before using the program; different read, write, add, and delete privileges will be assigned depending on the user.

2.3 Graphical User Interface

The GUI meets all the requirements specified in our *Functional Specifications* document. There are currently three main GUI forms in the BCA: start-up form, main form, and database form.

The start-up form displays one of two options upon program start up: Analyze sample, or view/modify database. A screenshot of the present start-up form is shown in Figure 3:

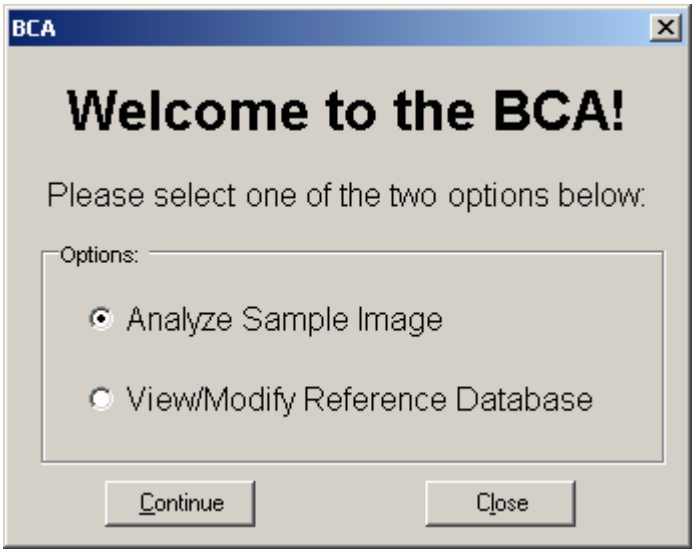

Figure 3: BCA Start-up Form

Presently, due to the information that needs to be gathered, users can only add to the database once the Analyze Sample Image Wizard has been run. A possible third option

for the future is adding records to the database directly, with prompts for users to enter the appropriate information.

Of the most use is the database form, which pulls the correct information from the database and displays them to the user. The picture box for displaying bacterial images is flexible, allowing for pictures of different resolutions and sizes. A screenshot of the database form is show in Figure 4:

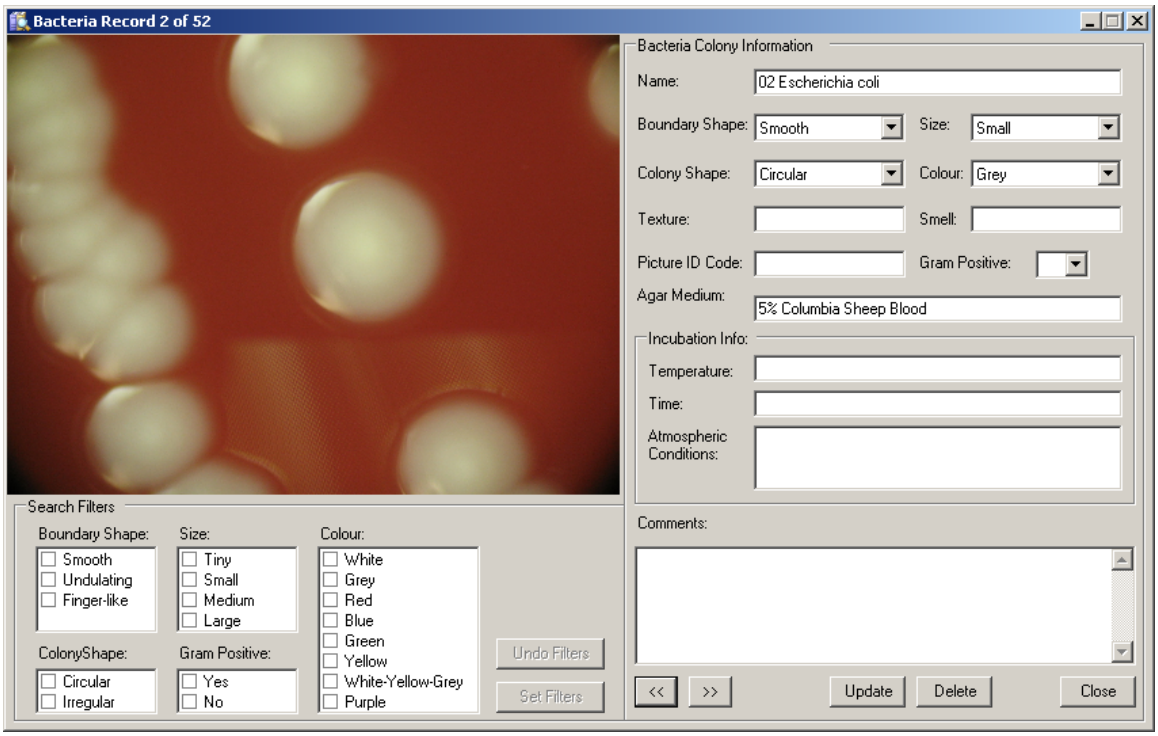

Figure 4: BCA Database Form

One of the most powerful features of the database form is the ability to set search filters. The filters may be set by the CC return values after the Analyze Sample Image Wizard has been run, or manually by the user on the database form. Currently, classifiers within each CC are OR-ed, while classifiers between different CCs are AND-ed. A preliminary design was done to allow the option of AND-ing and OR-ing between the CCs, but was deemed too confusing for the user. Thus, a future option would be a separate advanced filtering window with more powerful functions, such as AND-ing, OR-ing, and keyword searches.

Currently, the users scroll through each entry one by one. A future improvement would involve some sort of thumbnail preview containing the subset results. Clicking on a thumbnail item would show the full bacteria information on the form. An option of users to jump directly to a specific record number is also being considered.

The main form consists of a wizard that guides the user through the analyze sample image process. Currently, the user must perform these steps in order:

1.) Specify the location of the calibration image.
2.) Click on a point within the calibration image.
3.) Specify the location of the sample image.
4.) Crop a rectangle in the sample image that contains one bacterial colony.
5.) Draw a rectangle within the cropped bacterial colony.
6.) Select which tests to run.

We would like to eliminate steps 2, 4, and 5 completely. If steps 4 and 5 cannot be eliminated, it would be preferable to have a lasso tool for the user to draw around the colony instead of the more restrictive rectangle. Also, the steps should be rearranged so that step 6 becomes step 1. That way, other steps may be skipped, depending on which tests the user wishes to run.

3 Timeline

The original timeline of our project is shown in the Gantt chart in Figure 5. Although we followed the general structure of our timeline, we discovered that coding and testing the various modules took longer than we expected and significantly decreased our planned overall testing period. The updated timeline of our project is shown in the Gantt chart in Figure 6.

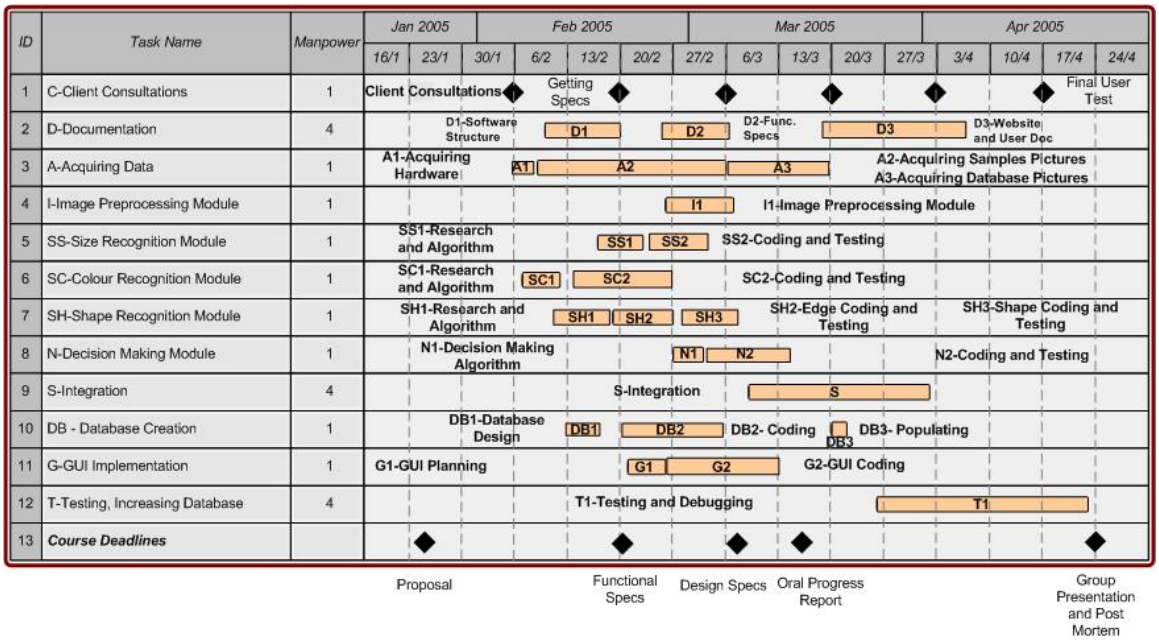

Figure 5: Original Gantt chart

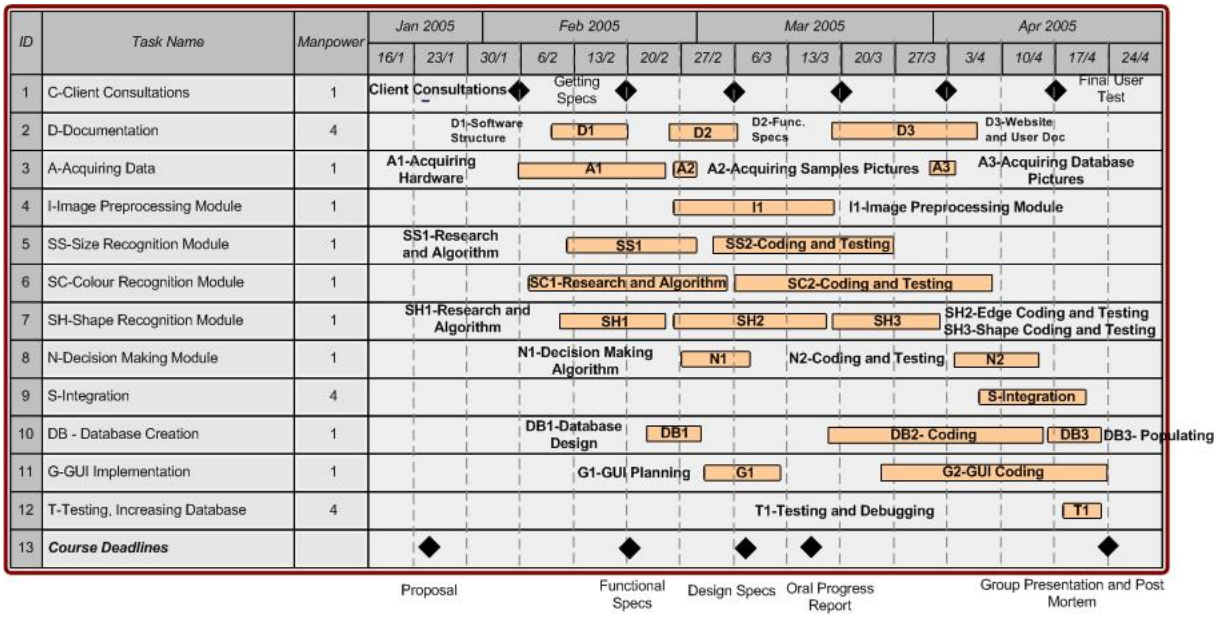

Figure 6: Updated Gantt chart

4 Budget

Because our project is primarily a software product, we were able to take advantage of educational licensing and programs available to us within the Engineering labs to keep our costs reasonable. The following table shown in Table 1 displays our estimated budget for the period of January – April 2005 as well as our actual costs for the same period. Unexpected costs are shown bolded and italicized. Overall, our actual costs are close to our estimated costs because we were able to save on other costs such as research expenses by borrowing materials from friends and professors. We were also able to borrow the lab equipment for acquiring samples from the lab in BCCDC at no cost.

Item	Estimated Market Price (CDN)	Our Estimated Cost	Our Real Cost
Research expenses (photocopying, printing)	$100.00	$100.00	$10.25
Textbooks on C++ and Visual Studio			**$127.98**
Developmental tools and software			
· MS Visual Studio .NET Academic Edition 2003	$135.00	$0.00	$0.00
· Matlab Student Version with image acquisition, image-processing, and fuzzy logic toolboxes.	$177.00	$0.00	$0.00
· Paint Shop Pro 9.0	$144.99	$0.00	$0.00
· ***CVS Advanced***			$12.00
Sony Cyber-shot 3.2 DSC-P72 mega-pixel digital camera	$300.00	$0.00	$0.00
Microscope by C&A Scientific	$135.00	$0.00	$0.00
Lab Petri Dishes 100 x 15mm (20/Pk)	$13.50	$0.00	$0.00
Travel expense for consultation meetings	$15.00	$15.00	$7.75
Website	$15.00	$15.00	$18.11
Consultation Lunches			**$24.04**
Contingencies	$20.00	$20.00	$0.00
Total:	**$1,055.49**	**$150.00**	**$200.13**

Table 1: Comparison of Estimated and Actual Budgets

5 Future Developments

5.1 Improvements on Current Image-Processing Modules

The pre-processing module that needs to be improved is the edge detection module. Because the outputs of this module are used by all other modules in the image-processing section, improving the reliability and accuracy of the edge image and edge sequence is a priority. The current centre finding is extremely accurate when the edge is accurate and free of noise.

Because all the classification modules rely on an accurate edge, much of the effort in future developments will be centred on creating a more reliable edge of the image. One method of accomplishing this is to use Active Contours instead of gradients. The idea behind active contours is to have a predefined contour superimposed on a gradient vector flow field map for an image. For each point on the contour, the fields exert a force on that point moving it towards the closest and strongest edge. The initial location of the contour does not matter since the field map will force the contour to converge on the edge after sufficient iterations.

Another method would be to create the gradients directly from the colour image. Currently the image is split into the red, blue, and green planes, and the 2 dimensional gradients are calculated from each plane. If a 3 dimensional gradient was directly calculated from the colour image, the edge may be able to pick up boundaries between colours such as red and purple, which is currently a problem in the BCA. Both methods have the potential to greatly increase the accuracy of the edge and improve the overall functionality of the BCA.

With an accurate edge detection module, the size, colony shape, and boundary shape modules will improve in accuracy automatically. The colour module will also improve in accuracy although lighting dependency is also a big factor in the reliability of colour. All of the modules can become more reliable by increasing the number of categories and making the distinctions between categories finer. Making the categories finer will allow the search algorithm to be more accurate in matching the candidates and return a smaller subset. Even with secondary categories taken into consideration, more categories will produce a more accurate and useful search functions.

Another major problem encountered was poor image quality due to the way the pictures were taken. A digital camera microscope with more diffuse lighting and multiple images of the same sample may improve reliability of the image processing. We also noted that the camera used was not focusing properly on all parts of the colony. If the edges in the image are in focus and sharp, the reliability of the edge module should increase significantly.

5.2 Additional Functionalities

5.2.1 Expanding the BCA

Future plans for the BCA include adding more classification criteria for bacteria identification. Possibilities include the texture, the 3D structure of the colony, and the growth rate of the colony. Additional CC's will increase the matching accuracy of the BCA and increase the usefulness of the BCA's browsing functions. Most of these new CC's can be categorized using similar image-processing methods currently implemented in our classification modules.

5.2.2 Increasing Flexibility of the BCA

The current BCA modules are calibrated for bacterial colony identification alone. However, the same modules (with different calibrations) can be used to identify a host of different single-cell organisms. A possible future development is to expand the GUI to include a module training screen that will allow users to set their own category parameters and thresholds for detection. The individual modules will be able to accommodate changing threshold values and expand the use of the BCA beyond the bacteria identification field.

A more image-processing-friendly calibration is needed since the current object does not return a consistent area for comparison. No further improvements are suggested for the centre detection but testing with incorrect centre points may yield useful results for the colony and boundary shape modules. More colony shapes should be added since circular and non-circular is a very limited range at the moment.

6 Personal Comments

6.1 Edward Goh

As CEO, I was responsible for contacting our client, David Chan, at the BC Centre for Disease Control. I found that coordinating a technical project with other people that are in a different field could add a lot of extra time to the timeline. The difficulties were always related to the difference in backgrounds and how explaining algorithms or system requirements required extra details and a few attempts before the correct information exchanged hands.

On the software side, I was in charge of the modules for size, centre finding, and colony shape. These properties were covered in ENSC489, which I had already completed, so I was prepared at the beginning of the project to design algorithms for the modules that needed to determine the features. However, familiarizing myself again with C++, image-processing techniques, and the OpenCV library required a lot of my time once I really started working on the modules. If I were to do a similar project, I would like to have much better software experience before starting or group members that are highly experienced with coding full projects in C++.

Overall, the project progressed well until near the end of the semester where we had some delays before we successfully integrated all of the modules. I believe we met our goal of presenting a proof of concept to our client that can be built upon in the future. I enjoyed working with my group members and team dynamics was never a serious problem.

6.2 Linda Wu

For the project, I was responsible for designing the Colour Module and testing the overall integrated system. Some of the challenges I faced include deciphering the OpenCV documentation and recalling the object-oriented programming concepts I learned a couple of years ago. Because C++ was a programming language I've never used before (although I do have experience with Java and C), I sometimes found programming to be very frustrating. Many of the reasons my module would not function as expected was due to small mistakes in syntax. Looking through the OpenCV documentation and learning to use the library effectively has also taught me the value of good documentation and the danger of assuming that good documentation is always available. Although the OpenCV library is very powerful, I do not think I used it as effectively as I would have if the documentation were more comprehensive.

On the management side, I was responsible for sending out the weekly agendas for meetings and chairing the meeting. The experience taught me the importance of having agendas for group meetings in order to accomplish everything we needed to do. However, I realized that as we worked together throughout the semester, formal agendas were not

as important. As our group met and worked together, we became more efficient and better able to keep the discussion on track, although having an informal agenda for every meeting was still very helpful.

I found the topic of this project to be very interesting. I have always had an interest in biochemistry and this project exposed me to a sector of engineering that I had not previously considered before: bio-imaging. I found that despite the challenges of the project, and the endless hours in front of my computer programming, I enjoyed the process of creating and testing a program. I found that I much preferred debugging software to debugging hardware, since I have notorious luck with hardware. Overall, I am very glad that I was able to be a part of this group and work on this project.

6.3 Josna Rao

For this project, I was responsible for designing the Edge Pre-processing Module and the Boundary Shape Module. I was able to familiarize myself with the library of OpenCV functions in order to efficiently use our time and avoid redesigning the wheel. The Edge Pre-processing module was fairly challenging because of the large variety of colonies our program had to deal with. Since colonies range from every sort of colour and shape, and since the background nutrient solution ranged in colours, our edge module had to be able to detect edges even if the colony colour could not be sharply define from the background colour. I found that doing the boundary shape module was somewhat easier due to the use of histograms.

Overall I thought our group was able to function well. Since our project already had fairly defined modules, we were able to split the work easily. One of the areas where we ran into difficulty was in the integration process where we underestimated the amount of time needed. Since integration took longer than expected, the amount of time for testing was reduced to the week before the demonstration. We were also somewhat pressed for time due to the weeks we spent searching for a project idea at the start of the semester and the time spent researching image-processing techniques.

Despite the lack of sleep involved in doing this project, I found the experience to be valuable and enjoyable. Our project was fairly advanced and in a field of research that is often carried out at the graduate level. Since I am interested in the bio-imaging field, this project gave me the perfect opportunity to see how I would feel about this type of research. My group and I have also gained many valuable skills, primarily because we were not familiar with image-processing techniques prior to the project. Overall, I am glad that we chose the project idea that we did.

6.4 Geeda Ip

I was responsible for designing and creating both the database and GUI for the BCA. Although all three of my co-ops were software based, I was a bit surprised at the amount of learning involved. I soon learned that managed extensions programming with C++ was quite unlike standard C++; this became evident during the integration phase, as I had to learn mixing managed and unmanaged code. Furthermore, the documentation provided by Microsoft in the MSDN library was a bit confusing; because most of the examples were given in Visual Basic or C#, I had to first gain a basic knowledge of these two languages. I eventually became more familiar with MSDN documentation towards the end of the project – how it was structured, what to ignore, what to expect, and where to go for addition help.

In setting up the database, I had to learn Microsoft Access and ADO .NET. Once the initial data structure was designed in Visio, the database was created in Access. Reading, updating, adding, and deleting data were done in code via ADO .NET. I had to quickly familiarize myself with ADO .NET concepts, such as datasets, data adapters, command objects, and XML schemas. Getting ADO .NET and Access to communicate took a very long time. I would often find that after hours of dissecting my code, the reason that the program was throwing an exception was because one tiny property in an Access table or the dataset was set incorrectly.

Since we decided to program in C++ using Visual Studio .NET 2003, I chose to take advantage of the Visual GUI designer and used .NET Windows Forms to create the GUI. This involved learning a new style of programming since event handlers and control properties had to be tediously set up. Of the 4000+ lines of code that I wrote, approximately 75% were dedicated to the GUI in some aspect or the other. However, I also realized that nobody really cared about the GUI because "it's just expected to work." Like the sound effects of a movie, the only time anyone notices it is when it's done improperly.

Because the solution and project files in Visual Studio .NET often required peculiar property settings and inter-dependencies in order to compile, I created a CVS repository for our code. Using WinCVS and an account set up at www.cvsdude.org, our group members were able to download our code, openCV library files, along with the database and sample data pictures, to any computer with Internet access. Furthermore, if Visual Studio .NET 2003 is installed, they can compile the code without any project setup knowledge. CVS was an invaluable resource, especially during the integration phase, as each member had to continuously update his or her code. Furthermore, each check-in is versioned, which proved useful since we could easily fall back on previous versions of our code after drastic changes.

In addition to my programming duties, I took the minutes for our weekly meetings. Action items were particular useful, as we could all refer to them prior to next week's meeting and see what needed to be done. They became useful over time as references to

issues discussed and decisions made. I also registered and maintained our website, www.adarza.com. In that process, I learned a bit about domain setup, account setup, and HTML.

Overall, I found this project to be quite interesting, albeit time consuming. The skills I gained from .NET programming are not taught at SFU and will be advantageous to me in the industry. I also believe that our group worked together quite harmoniously and were supportive and helpful of each other. We all worked hard, but managed to retain our sense of humour, even during the final days before the demo. I am pleased to have worked on this project with my group members.

7 Conclusion

Although the current BCA does not meet our *Functional Specifications*' accuracy goals, we did establish that image-processing methods could be very useful in the field of bacterial colony identification. We feel that, given more time, we will be able to raise the reliability of our project to above the accuracy levels specified in our *Functional Specification*. The BCA has to potential to become more than just a bacteria identification system, with possible expansions into parasite identification and cell identification.

Despite being constantly pressed for time, we are satisfied with the overall product we were able to produce at the end of 13 weeks. This project has introduced all of us more intimately into the field of image-processing and bio-imaging, and given us a glimpse of what it would be like to work in these fields.