Graphicizer: Image Editing and Conversion Tool

November 22, 2008

Contents

List of Figures

Abstract

"A picture is worth a thousand words." This simple proverb conveys that complex information or ideas can be expressed with a single image. This cognitive process of seeing an image and understanding its meaning is being applied to visualization, which aims to present large amounts of data into a single image that can be absorbed and understood quickly. Visualization is a powerful analytic and reporting tool that empowers businesses to take control of their data and quickly turn it into actionable information. Image processing is the art and science of manipulating digital images. It stands with one foot firmly in mathematics and the other in aesthetics, and is a critical component of graphical computer systems. If you've ever bothered with creating your own images for Web pages, you will no doubt appreciate the importance of Photoshop's image manipulation capabilities for cleaning up scans and clearing up less-than-optimal images.

If you did any image processing work in JDK 1.0 or 1.1, you probably remember that it was a little obtuse. The old model of image data producers and consumers is unwieldy for image processing. Before JDK 1.2, image processing involved MemoryImageSources, PixelGrabbers, and other such arcana. Java 2D, however, provides a cleaner, easier to use model.

Today the technology is fast advancing and it becomes the duty of a professional is to utilize it properly and make it available to the masses. Everything is computerized online. For that purpose fast, powerful and smaller computers are developed for commercial and scientific applications. The computer industry focuses on the era of cost, flexibility, reliability and other such immaterial considerations. Our aim is to make the conventional systems computerized with maximum security, thus making it possible for the organizations to serve their customers well.

The system entitled "Graphicizer" is application software, which aim at the development of Image Processing System and it is developed by using JAVA SWINGS technology and related technologies.

Graphicizer is mainly concerned with managing the operations defined over an image. The main objective of this project is to develop a system that is less complex and can be used within any domain area.

This project deals with the process of selecting an image, performing transformations such as blurring, sharpening etc. on the images, and storing the images on the system.

The system developed is a system utility so it can be used by any user. This system is not tied to any particular company nor married to any firm.

This report describes the various prospectuses of developing and using the system along with the technical aspects of software engineering.

Chapter 1

Introduction

Images make a lot of difference as they influence a person's decision in many cases. With the development of computer technology, many things have changed. Today there is enough and more technology available to improve pictures that have flaws and to add special effects to existing pictures. We are equipped with the required technology and software for Image Editing / Enhancement that meet industry standards. Our Image Editing / Enhancing services enable you to correct any deficiency in your original images and make them more presentable. The services we provide include:

- Correcting photographs which are dull or dark
- Transformations on image such as Blurring, Sharpening, Image Negative, Brightness, Edge Detection , Contrasting
- Changing color to Gray Scale Conversion
- Image format conversion

Problem: To begin work on this project, you will need to gather several two-dimensional found images and materials (such as old photographs, newspaper clippings, tickets, loose fabric, graphics, printed words, poetry, sheet music, etc.). If you'd like, you may experiment with three-dimensional objects.

The content (what the work is about) of the image(s) you create is up to you. For example, the image(s) you create might: be autobiographical in nature; be suggestive of a well-known personality, story, or place; be a response to a social or political event; or be related to formal concerns. Consider that if you want "art out of the computer" you have to "put art into it." Neither the scanner nor Photoshop alone will produce an interesting work of art! You may start with an idea and then look for things to convey that idea, or you might let the "stuff" you collect suggest an idea to you.

Remember...

- Images can be symbolic. They can evoke emotional responses such as irony, humor, nostalgia, or dread.
- Images can be pictorial. A combination of images and objects can illustrate ideas that dominate our thinking, showing us in retrospect.
- Images can be metaphorical. They can serve as substitutes for other objects, ideas or experiences that are comparable in some symbolic way.
- Images can be abstracted. Selected photos can be collaged to provide balance and harmony.
- Above all, images trigger personal, highly individualized responses that might depict or remind us of ourselves, our problems, our joys.

1.1 Problem Identification

Whenever we want to work with images, we quickly start searching for an Image Editing Tool. There are already lots of tools available in the market such as Adobe Photoshop, Image Viewer and all. Suppose if we want to perform simple functions on images such as Blurring, Sharpening etc., we generally wont prefer loading such big applications which are too complex to use. A naive user could get fainted while seeing such complex applications and will hesitate to use it again. It requires a lot of effort to learn the applications working.

1.2 Available Solution

Lots of applications available for Image editing or Picture Editing. Some of them are freely available and some are paid. Various available products are:

- Adobe Photoshop Express
- Picnik
- FotoFlexer
- Aviary
- Splashup
- 72Photos
- Scrapblog
- Pixelmator
- and many more...

We here provide a small and less(no) complex software that enables a naive user to work with fun. The user need not have any prior knowledge of any kind of imaging tool to use this software. The software is user friendly. Software allows the user to find out enhanced image from the different method.

1.3 Product Perspective

1.3.1 User Perspective

Graphicizer is a free image editor with many of the features of a professional product. You can edit images from a variety of formats; it contains an extremely user-friendly interface enabling you to work intuitively. Along with the usual assortment of file operations such as Copy, Move, Delete, and Rename; adjust colors in photos; apply filters such as Hue, Negative, Colorize, Emboss; transformations; Perspective and Skew; blur images; sharpen images; image resizing; setting an image as the desktop wallpaper. Also, it contains enhanced brushes and painting options.

1.3.2 Business perspective

The pressure is mounting for technology organizations to measure the real ways their departments are adding value to the business at every level. Business organizations are increasingly dependent on the electronic delivery of services, irrespective of type or size of organization, and require high quality information systems (IS) services which can adapt to business and user requirements as they evolve.
The application area of "Graphicizer" is wide in range. The "Graphicizer" can be used in educational institution, in scientific research institution, in medical science, military application, film industry etc. This project can also be used in mobile industry to provide good quality of picture or noiseless image.

1.4 Feasibility Study

For all new systems, the requirement engineering process should start with a feasibility study. The input to the feasibility study is a set of priminary business requirements, an outline description of the system and how the system in intended to support business processes.

1.4.1 Technical Feasibility

A feasibility study is short , focused study that aims to answer a number of questions:

1. Does system contribute to the overall objective of the organization?

The user need not have any prior knowledge of any kind of imaging tool to use this software. The software is user friendly. Software allows the user to find out enhanced image from the different method.

2. Can the system be implemented using current technology and within given cost and schedule constraints?

The tool is made using JAVA which is platform independent language. the tool can be run on any platform using new and old technologies. The tool is

also feasible in terms of cost. The softwares and tools used for the development are quite freely available in the market.

1.5 Software Requirement Specification

1.5.1 Introduction

Scope

This specification establishes the functional, performance, and development requirements of a Software application for displaying and editing images.

Definitions, Acronyms and Abbreviations

CSCI Computer Software Configuration Item
IDO Image Data Object
IEI Image Editing Interface
ILG Image Layout Generator
ISP Image Save and Print
GUI Graphical User Interface
SRS Software Requirements Specification

1.5.2 Background

In the past and today also, lots of Picture editing tools had came into the market. Each and every product has their own benefits and are scalable to the needs of users. The very basic software in use by users is Paint in windows. It provides you with lots of functions and quite easy to use. However it doesn't provide any method to do transformations on image. Other tools such as Adobe photoshop, Fast stone viewer, Picnik, 72photos etc. provides features for such transformations having 10 to 100's of filters support.

1.5.3 Overall Description

The Graphicizer is a very cool image-handling tool that lets you load in images, manipulate them, and save them back to disk. It'll convert image formats too. You can use the Graphicizer to load in JPEG, PNG, or GIF images and then store them in either JPEG or PNG format. When an image is loaded, you can convert it to an embossed image with the click of a button, or you can sharpen it, brighten it, blur it, or reduce its size all using Java. You can even undo the most recent change. Images are commonly used throughout software engineering and computer science to present structured information in a form that is easy to understand. Many forms of information can readily be presented as an Image, including structural and behavioral views of a software system. The software described in this specification consists of an application for creating, viewing, editing, laying out, saving and printing the kinds of images.

1.5.4 Environmental Characteristics

Hardware

RAM: 128 MB
Memory: 2.5 GB

Tools Used

While going through the whole process of development we have used the following tools for various purposes. The tools used are:

- **JDK 5.0 Kit**:
  This is the java environment kit.
- **JCreater**:
  This tool is used as JAVA editor for coding.
- **Latex**:
  Latex is used for documentation.
- **Visual Paradigm tool**:
  This tool is used for creating Class diagrams, Sequence diagrams, UML diagrams.
- **Dia**:
  This tool is used for creating Flow Charts, Data Flow Diagrams and Entity Relationship Diagrams.

People

Two Peoples:

1. Tarun Jain
2. Abhishek Verma

1.5.5 Functional Requirement

This section describes the functional requirements for each of the four CSCIs: Image Data Object CSCI (IDO), Image Editing Interface CSCI (IEI), Image Layout Generator CSCI (ILG) and Image Save and print CSCI (ISP).

Image Data Object (IDO)

The IDO CSCI shall store the current image in memory, and provide a functional interface to other CSCIs to access and make changes to the current image. It shall also be responsible for checking the Validity of the current image.

The main data used by the Graphicizer is a representation of the current image. IDO shall hold all data about the current image in a suitable data structure, and provide information about the image to the other CSCIs.

For the image as a whole, IDO shall store the filename that this image was last saved to (be any). Each image has a textual label that uniquely identifies it. The information about the size, location and appearance of image can be stored as either the attributes, or as type information.

Image Editing Interface (IEI)

The IEI CSCI shall provide a user interface for viewing and editing images. IEI shall provide a view of the current image, along with the ability to perform transformations like Embossing, Blurring, Black and White and all. All user interactions with the software shall be handled by the IEI. Hence IEI shall also provide user commands for opening and saving to files, for navigating the file system, and for generating and modifying layouts.

IEI shall not store any data about the current image. IEI shall use the IDO to store the current image, including layouts and appearance information, and to check the current image is valid. IEI shall use the ILG to generate layouts prior to displaying and image that do not complete layout information. IEI shall not directly interact with the operating system for access files; all access to the felsite shall be via ISP.

IEI shall store information about the past sequence of edit actions, so that it can provide a multi-level undo. IEI shall allow the user to undo an entire sequence of edits, back to the last time the image was saved to a file. IEI shall store a list of recently opened files, to allow the user instant access to these files.

Image Layout Generator (ILG)

The ILG CSCI is responsible for automatically generating a layout for the current image. It doesn't store any data about the current image. It used the IDO to store the current image, including layout information, and for checking the current image is valid.

The ILG may need to store temporary information about the layout being generated, according to the algorithm used. ILG shall keep a copy of and layout for the current image that it overwrites, so that old layouts can be retrieving by the user via the undo command.

Image Save and Print (ISP)

The ISP CSCI is responsible for opening image file, saving the current image, and printing the image. ISP is responsible for all interactions with the file system, including checking whether a file exists, and moving between directories.

ISP shall support one image format chosen from the following: GIF image, JPEG image, or PNG image. The image saved shall correspond as closely as possible to the appearance of the image displayed by IEI.

ISP shall not store any information about the current image. All data read in from image file shall be passed to IDO to be stored.

1.5.6 Non Functional Requirement

The Graphicizer is broadly divided into four discrete modules.

- Image Data Object (IDO)
- Image Editable Interface (IEI)
- Image Layout Generator (ILG)
- Image Save and Print (ISP)

Image Data Object

The IDO CSCI shall store the current image in memory, and provide a functional interface to other CSCIs to access and make changes to the current image. It shall also be responsible for checking the Validity of the current image.

The main data used by the Graphicizer is a representation of the current image. IDO shall hold all data about the current image in a suitable data structure, and provide information about the image to the other CSCIs.

For the image as a whole, IDO shall store the filename that this image was last saved to (be any).Each image has a textual label that uniquely identifies it. The information about the size, location and appearance of image can be stored as either the attributes, or as type information.

1. **Graphicizer External interface**:
   The Graphicizer IDO provide following functions for use by other CSCIs:-
   - new(): Allows you to select a new image
   - open(): Allows you to open the file dialog for selecting a new image
   - close(): Allows you to close the application
   - save(): Allows you to save an image
   - saves(): Allows you to save the file with new name or to a different location
   - undo(): Allows you to back your processes
   - redo(): Allows you to re do your work/process
   - locktoolbar(): You can lock your toolbar so as to protect it from performing some other transformations
   - blurimage(): Given an image, perform blur operation to make the image blur. The visibility will get blurred
   - sharpenimage(): given an image, perform sharpen operation. This will make the image more brighter and sharpened.
   - choosecolor(): various color transformations can be performed on the image using this feature.

Image Editable Interface

The IEI CSCI shall provide a user interface for viewing and editing images. IEI shall provide a view of the current image, along with the ability to perform transformations like Embossing, Blurring, Black and White and all. All user interactions with the software shall be handled by the IEI. Hence IEI shall also provide user commands for opening and saving to files, for navigating the file system, and for generating and modifying layouts.

IEI shall not store any data about the current image. IEI shall use the IDO to store the current image, including layouts and appearance information, and to check the current image is valid. IEI shall use the ILG to generate layouts prior to displaying and image that do not complete layout information. IEI shall not directly interact with the operating system for access files; all access to the felsite shall be via ISP.

IEI shall store information about the past sequence of edit actions, so that it can provide a multi-level undo. IEI shall allow the user to undo an entire sequence of edits, back to the last time the image was saved to a file. IEI shall store a list of recently opened files, to allow the user instant access to these files.

- IEI Internal data:

  IEI shall store information about the past sequences of edit actions, so that it can provide a multi-level undo. IEI shall allow the user to undo an entire sequence of edits, back to the last time the image was saved to a file. The saved list of edit actions shall be cleared each time the current image is saved.

- IEI External Interface:

  The IEI CSCI uses the functions provided by IDO, ILG and ISP to provide information about the current image, to generate layouts and to open, save and print the current image.

Image Layout Generator

The ILG CSCI is responsible for automatically generating a layout for the current image. It doesn't store any data about the current image. It used the IDO to store the current image, including layout information, and for checking the current image is valid. The ILG may need to store temporary information about the layout being generated, according to the algorithm used. ILG shall keep a copy of and layout for the current image that it overwrites, so that old layouts can be retrieving by the user via the undo command.

- ILG External Interface:

  The ILG provide the following functions for use by other CSCIs:-

- layout(): completes a layout for the current image, preserving any location information already defined. Returns true if it successfully generates a new layout, false otherwise.
- newlayout(): generates a new layout for the current image, discarding any existing location information defined. Returns true if it successfully generates a new layout, false otherwise.
- undolayout(): restores the previous layout. Returns true if it successfully restores a layout, false otherwise.

Image Save and Print

The ISP CSCI is responsible for opening image file, saving the current image, and printing the image. ISP is responsible for all interactions with the file system, including checking whether a file exists, and moving between directories. ISP shall support one image format chosen from the following: GIF image, JPEG image, or PNG image. The image saved shall correspond as closely as possible to the appearance of the image displayed by IEI. ISP shall not store any information about the current image. All data read in from image file shall be passed to IDO to be stored.

- ISP Internal Interface:

  ISP shall not store any information about the current image. All data read in shall be passed to IDO to be stored in memory.

- ISP External Interfaces:

  The ISP CSCI provides the following functions for external use by other CSCIs:-

  - open(filename): given a valid filename in the current directory, returns true if it succeeded in loading a new image, false otherwise.
  - save():- save the current image to the file from which it was read. Returns true if the operation was completed successfully, false otherwise.
  - saveas(filename):- given a string, saves the current image with the given name in the current directory.
  - exists(filename): given a string, returns true if the given string is the filename of an existing file in the current directory, false otherwise.

Chapter 2

Project Planning and Scheduling

2.1 Project Planning

The project has been divided into phases based on the development stages and implementation time.

2.1.1 Phase 1

In the first phase the decisions about the project selection were taken. After investing all the factors like time, availability of tools, resources,& man power; we decided to work on small project that makes use of Software Engineering along with the some handful experienced programming language. A rough design and idea was documented to start work with.

2.1.2 Phase 2

Next, we start preparing Software Requirement Specification Document. Feasibility study and requirements were gathered. The detailed design of the system with DFD and Class Diagram was made. The SRS was submitted to TA's in the Lab.

2.1.3 Phase 3

In this phase, we begin the coding process. We identified four modules and started the process simultaneously. Basically the coding regarding the making of Graphical User Interface was done in this phase.

2.1.4 Phase 4

Testing was our next milestone. We unit test the modules using different tools as well as did the manual testing.

2.1.5 Phase 5

Finally, the correctness of codes and proper functioning of all the modules were constantly tested. In this phase we check all the modules separately for their correctness and integrity.

2.2 Process Model

We have taken here Evolutionary Model to develop the software tool.

2.2.1 Why?

The project developed is quite small. Evolutionary development is based on the idea of developing an initial implementation , exposing this to user comment and refining it through many versions until an adequate system has been developed. It fulfills all the aspects which are required during the development process. There are times when the requirements of a problem are reasonably well understood when works flows from communication through deployment in a reasonably linear fashion. We have taken sequentially approach to develop the software project. That is the reason that's why we have used evolutionary model.

2.2.2 How?

Initially we have collected the different modules, described it and going to the high level step by step. We have collected information about each module, collects requirements for those modules separately, designing, coding and unit testing and every step followed of evolutionary model.

Specification, development and validation activities are interleaved rather than separate, with rapid feedback across activities. An Evolutionary approach to software development is often more effective than the waterfall model approach in producing systems that meet the immediate needs of customers. The advantage of a software process that is based on evolutionary model approach is that the specifications can be developed incrementally. As users develop a better understanding of their problem, this can be reflected in the software system.

2.3 Time Box Scheduling

In project management, a schedule consists of a list of a project's terminal elements with intended start and finish dates.
The project was undertaken according to the following schedule:

WORK DONE	START DATE	END DATE
PHASE 1	08/08/2008	22/08/2008
PHASE 2	23/08/2008	05/09/2008
PHASE 3	06/09/2008	26/09/2008
PHASE 4	27/09/2008	10/10/2008
PHASE 5	11/10/2008	24/10/2008

2.4 Responsibility Distribution

The project has been done in the group of 2 members.

1. The requirement specification and designing is done by Tarun Jain.
2. Modules coding was done by Abhishek Verma.
3. Testing and Code improvement was done with collective effort.

Chapter 3

Design

3.1 System Design

There are essentially four main functional areas, which correspond to the four CSCIs:

- **Image Data Object (IDO):** This CSCI shall provide a memory resident data structure to hold the current image, which shall be used by the other CSCIs.
- **Image Editing Interface (IEI):** This CSCI shall provide a Graphical User Interface (GUI), to allow the user to view and edit images. If the image to be displayed contains layout information, the IEI shall use this information to arrange the image on the screen; otherwise IEI shall call IGLG to provide a new layout. The GUI shall provide functions to the user to add or remove images, reads in JPG, PNG, or GIF files, saves image in JPG or PNG format, Embossing images, Sharpen images, Blur images, Brighten images, and Black and White image. This CSCI shall not store image in memory. It shall call IDO to access and update information contained in the current image. It shall call ISP for opening and saving files.
- **Image Layout Generator (ILG):** This CSCI shall lay out an existing image using the aesthetic conventions of the appropriate application. ILG shall call IDO to access information about the current image, and to store layout information as part of this image.
- **Image Save and Print (ISP):** This CSCI shall be responsible for opening image files and for saving the current image.

3.1.1 Data Structure

1. **BufferedImage**:- for storing images

The BufferedImage subclass describes an Image with an accessible buffer of image data. A BufferedImage is comprised of a ColorModel and a Raster of image data. The number and types of bands in the SampleModel of the Raster must match the number and types required by the ColorModel to represent its color and alpha components. All BufferedImage objects have an upper left corner coordinate of (0, 0). Any Raster used to construct a BufferedImage must therefore have minX=0 and minY=0.

2. **BufferedImage[]**:- array of type bufferedImage for keeping track of last five filteredImages

3.1.2 Algorithm

1. **Mean filter(bluring)**:-
   - Considers each pixel in the image
   - Collect the pixel values of the 8 neighboring pixels.
   - Find the mean intensity value of the 8 neighboring pixel including the intensity of the pixel considered.
   - The intensity of the pixel is set to the calculated mean value obtained from step 3.
2. **Laplace filter(sharpening)**:-
   - Considers each pixel in the image
   - Collect the pixel values of the 8 neighboring pixels.
   - Here the brightness values of the neighboring pixels is subtracted from the central pixels. In convention these weights are to be multiplied by pixels surrounding the central pixel and this sum is normalized by dividing by the sum of the positive weights
   - The intensity of the pixel is set to the calculated mean value obtained from step 3.
3. **Image Negative**:-
   - Considers each pixel in the image
   - Collect the pixel values of the 8 neighboring pixels.
   - Here the brightness values of the neighboring pixels is subtracted from 255.
   - The intensity of the pixel is set to the calculated mean value obtained from step 3.
4. **Brightening**:-
   - Considers each pixel in the image

- Collect the pixel values of the 8 neighboring pixels.
- Here the brightness values of the neighboring pixels is increased by some factor.
- The intensity of the pixel is set to the calculated mean value obtained from step 3.

3.2 Flow Chart

Refer to Figure:3.1

3.3 Dataflow Diagram

Refer to Figure:3.2 & Figure:3.3

3.4 Class Diagram

3.5 System Sequence Diagram

3.5.1 Scenario

For each user, the user of the program has a pile of image cards with the image name on it. The image card contains operations, supervised by the user. The cards are iteratively placed on the buffer until each image card is supervised. If the user wants, he or she may move or remove cards. Each time user performs operations on an image, the Image Statistics is updated. The image statistics enables the user to see how many operations has been performed.

3.5.2 Use Case Diagram

While the customer specification is quite detailed , the attempt to create a use case diagram shows that some basic information is only implicit or even missing. We donot provide use case for backing up the system's data and for starting and shutting down the system because these activities are outside the system's scope and donot pertain to the services it provides.

Also, information about the user of the system in not given explicitly. According to our interpretation, there exists only one kind of user, "Image Editor", who works with the system.

Refer to Figure:3.4

- Edit Image: Provides functionality for assigning users with images
- Manage Image Plans: Covers the functionality concerning whole image operations (like contrast, sharpening, blurring, persistemnt management etc.)

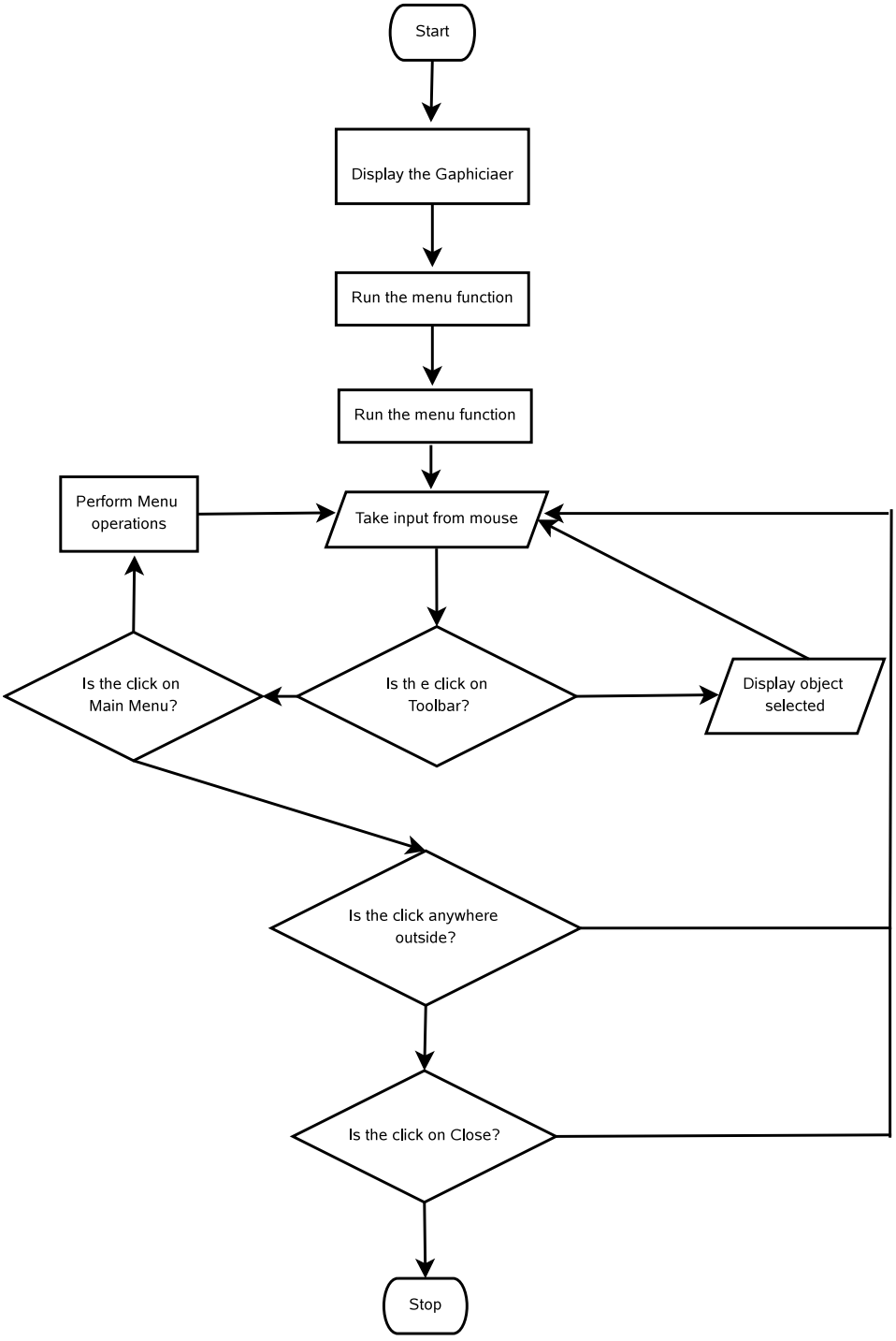

Figure 3.1: Flow Chart

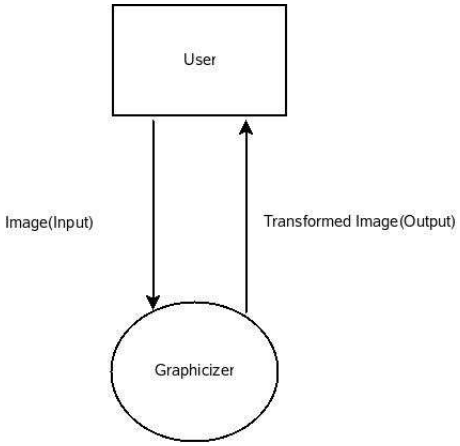

Figure 3.2: Context Level DFD

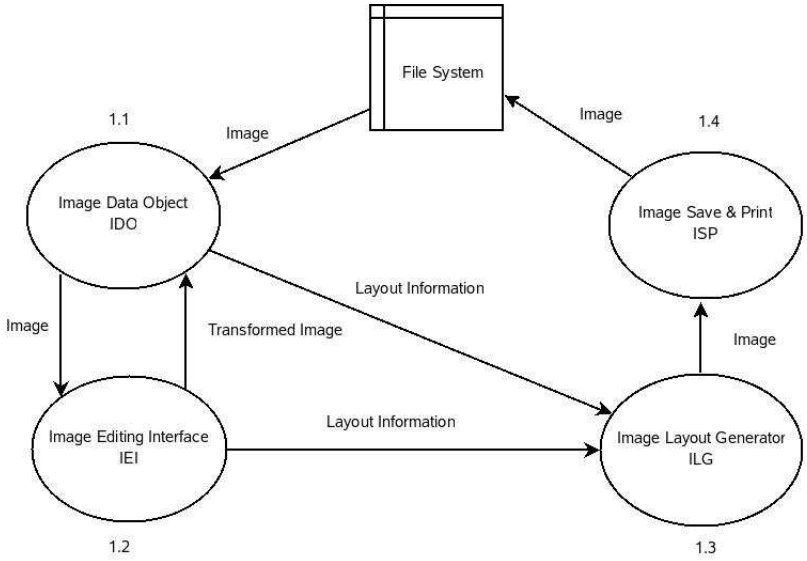

Figure 3.3: Level 1 DFD

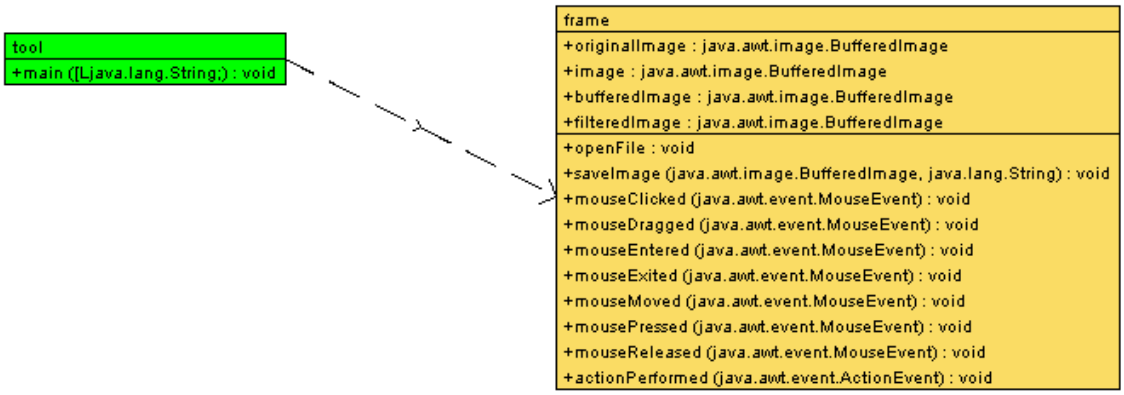

Figure 3.4: Class Diagram

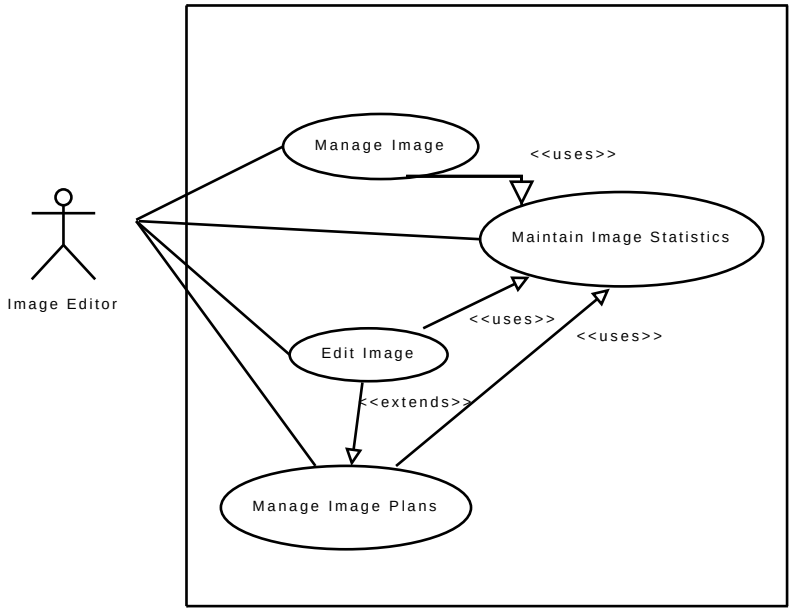

Figure 3.5: Usecase

- Manage Image: Provides functionality for adding and removing images
- Maintain Image Statistics: Updating the changes and presenting them to the user

Description of User

The user has not to deal with sensitive data and thus needs no special edit permissions.

- Number: Usually only one user at a time
- Experience: Normal user/Naive User
- Location: Anywhere

3.5.3 Sequence Diagrams

Edit Image

Refer to Figure:3.5

Sequence diagram for the updates

Refer to Figure:3.6

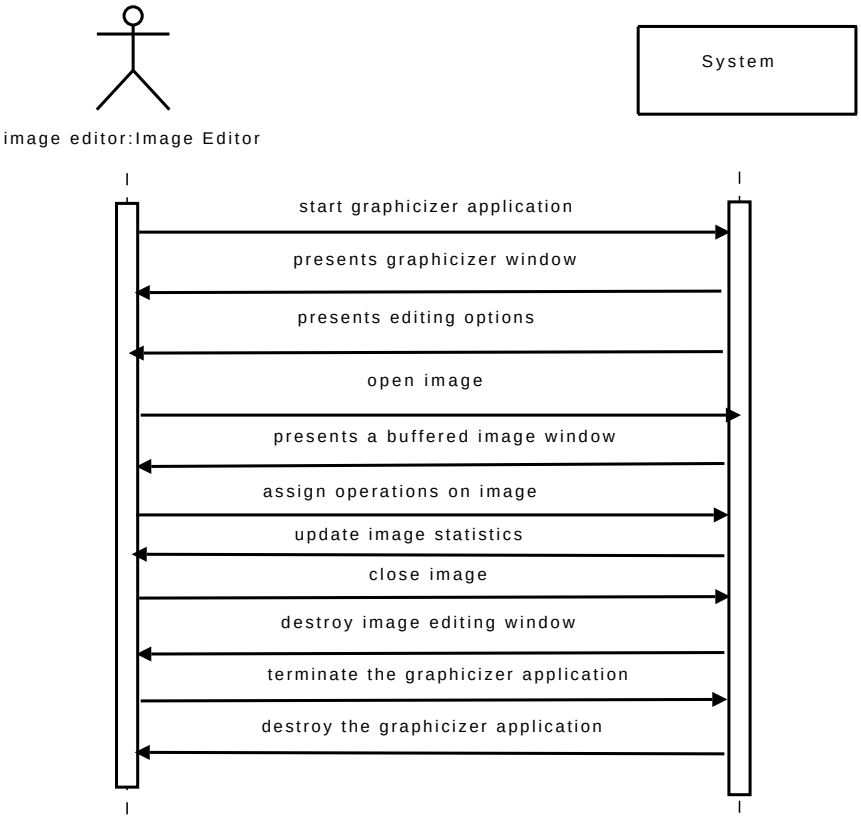

Figure 3.6: Sequence Diagram: Edit Image

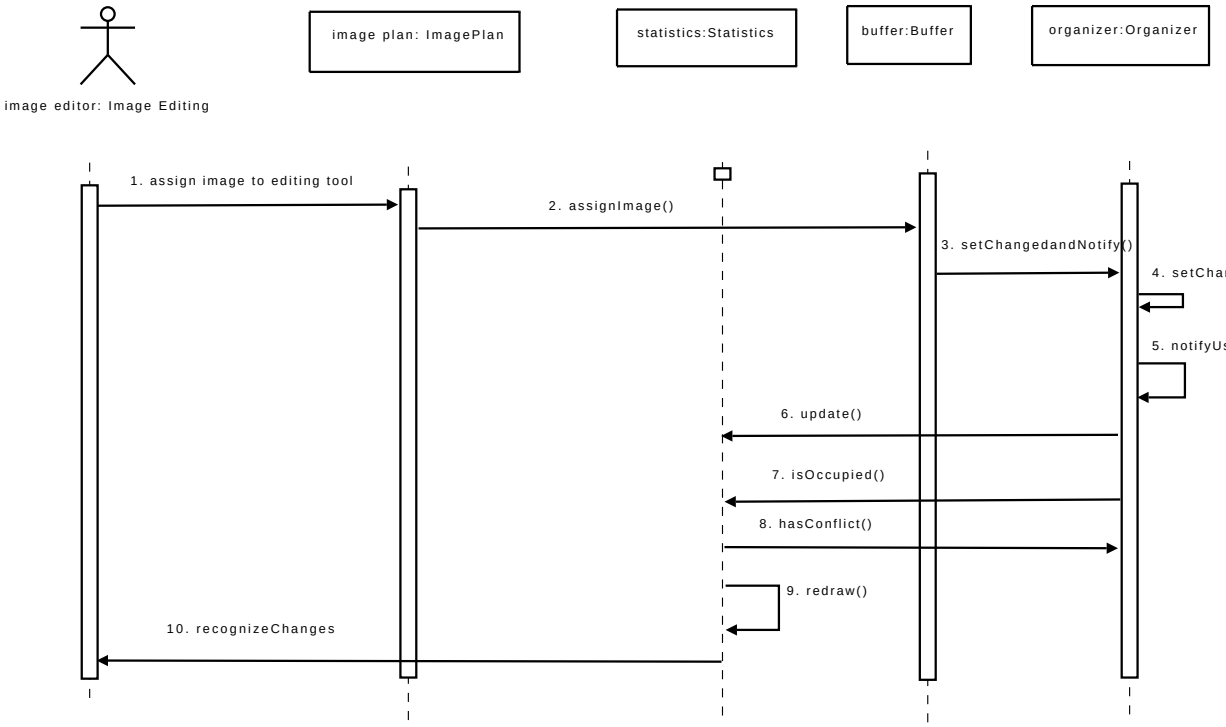

Figure 3.7: Sequence Diagram for update

Chapter 4

Coding

Chapter 5

Testing

5.1 Unit Testing

5.1.1 Test

- Case No. 1.1
- Title: Unit Test(UT)
- ProgName: Graphicizer
- Author: T&A
- Date: 06/11/2008
- Background: (Objectives, Assumptions, References, Success Criteria)

Validate that the UT will show the overall success and failure by generating test cases.

5.1.2 Test Results

- Case no. 1.1
- ProgName: Graphicizer
- Author: T&A
- Date: 06/11/2008
- Total Code lines covered: 173 out of 193
- Percentage of code covered:90%
- Test cases Success rate: 62%
- Test cases failure rate: 38%

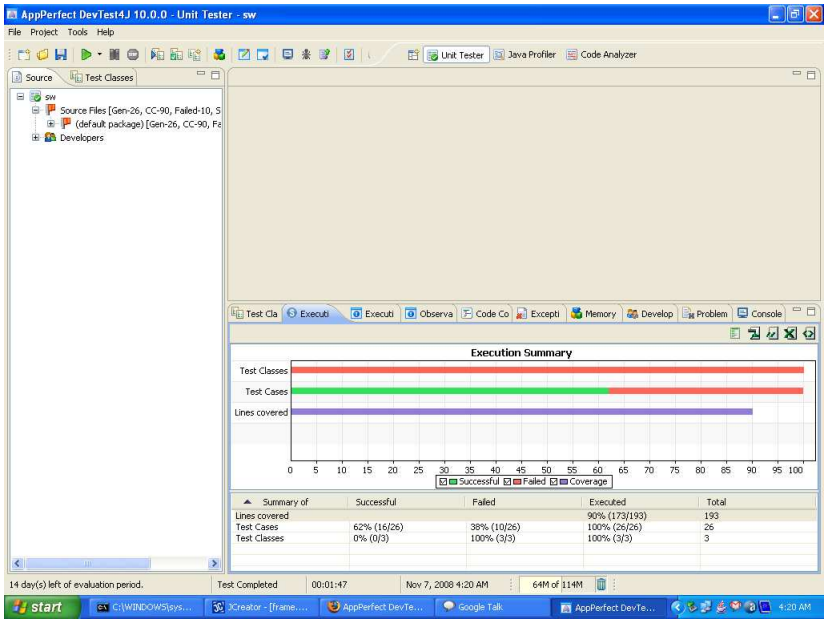

Figure 5.1: Unit Testing

5.2 Testing Snapshot

Refer to Figure 5.1

5.3 Snapshots of final project

Refer to Figure 5.2 to 5.10

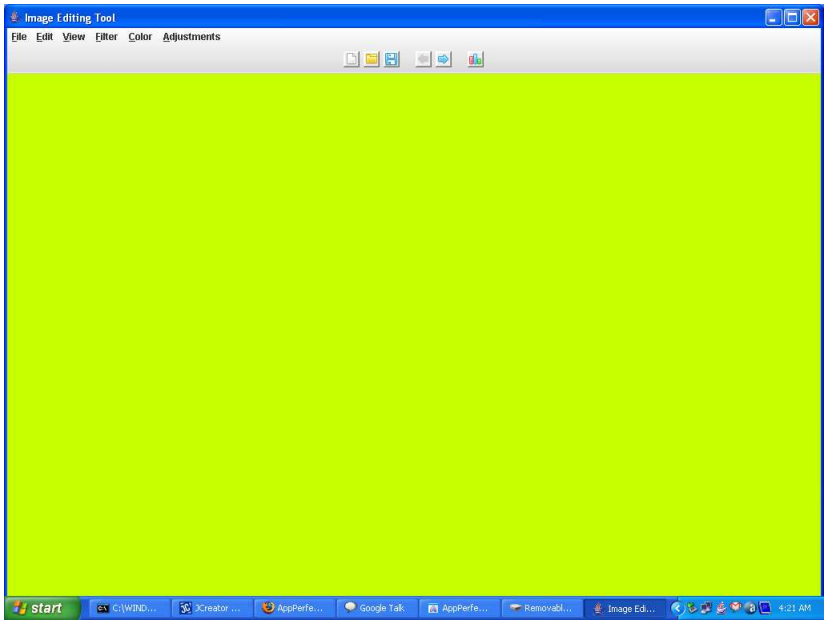

Figure 5.2: The Graphicizer Window

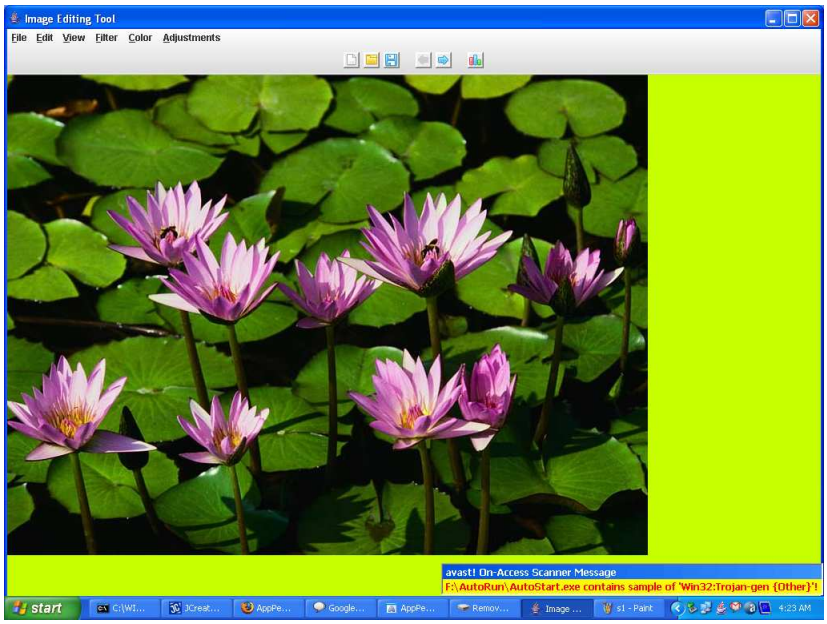

Figure 5.3: Original Image

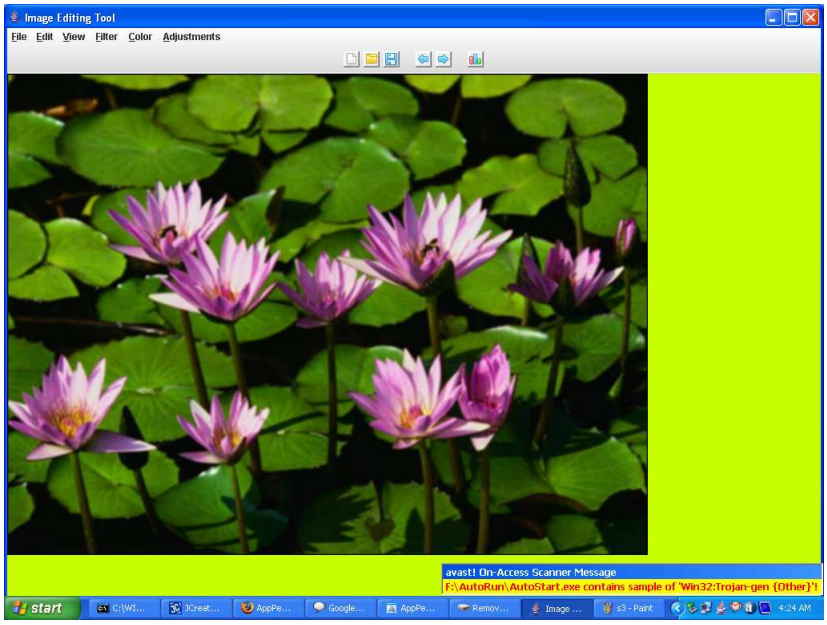

Figure 5.4: Blurred Image

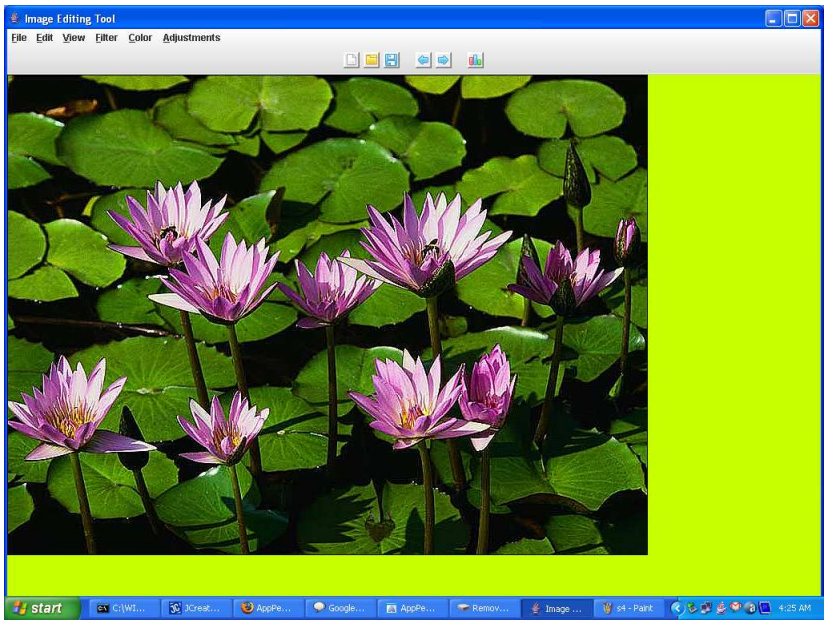

Figure 5.5: Sharpened Image

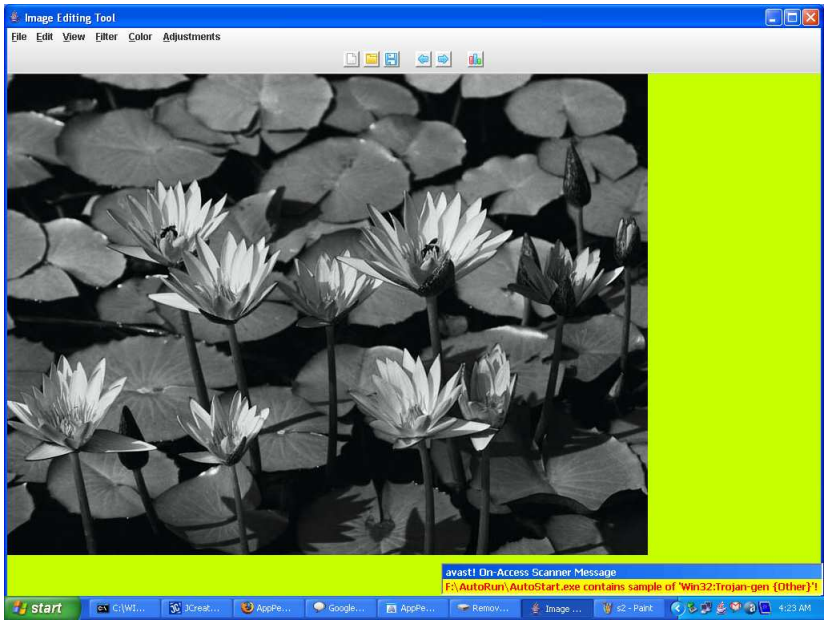

Figure 5.6: GrayScale Conversion

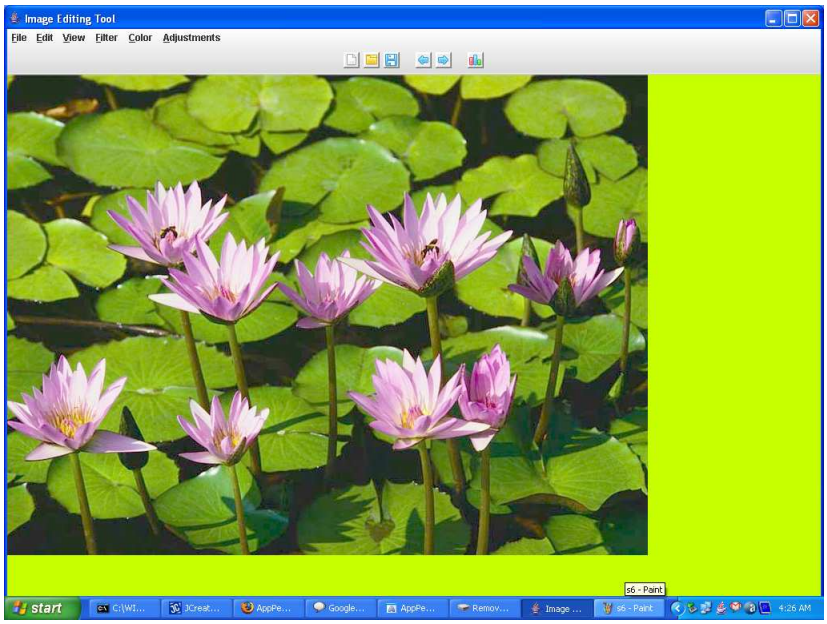

Figure 5.7: Brightnened Image

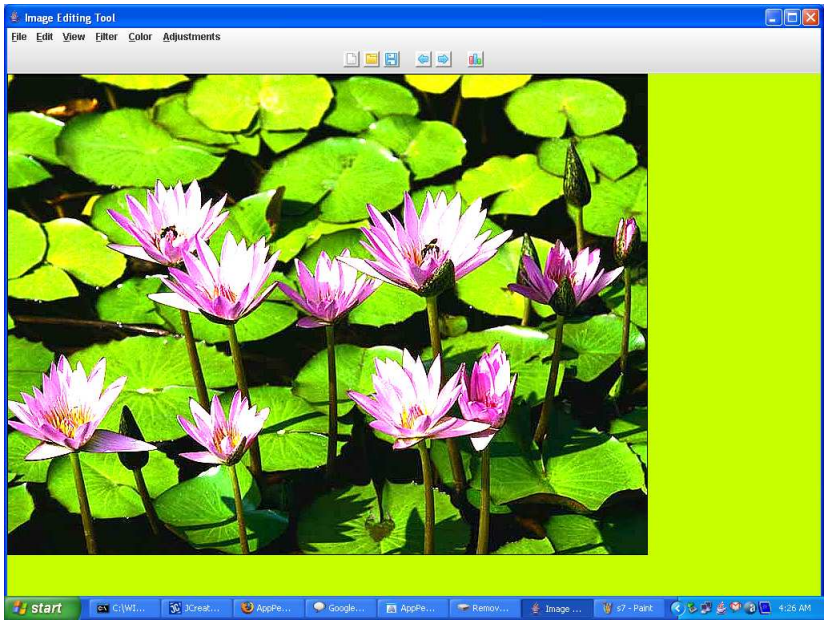

Figure 5.8: Contrast Image

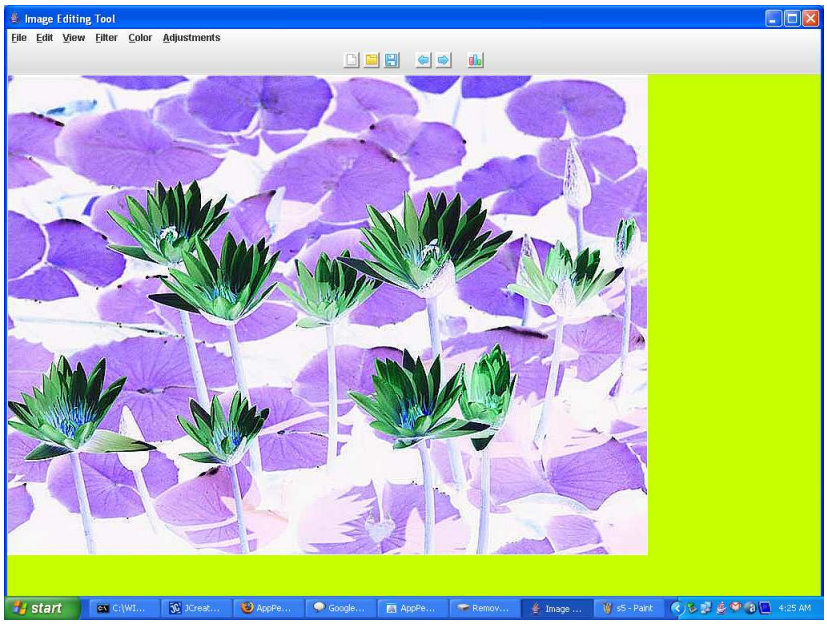

Figure 5.9: Image Negative

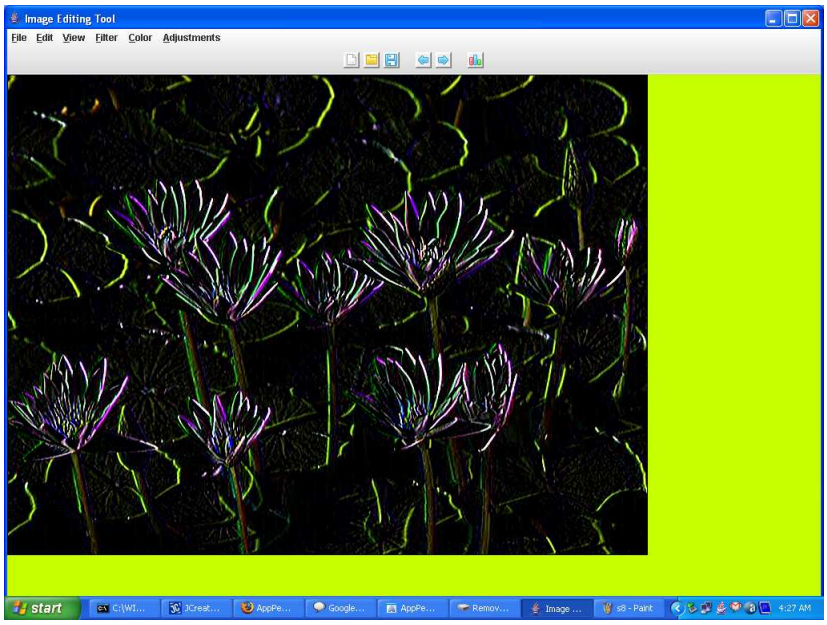

Figure 5.10: Edge Detection

Chapter 6

Future Work and Conclusions

6.1 Product Evaluation

A product is a term for any item that has been manufactured and is useful to you. You are a consumer when you buy it or use it. Evaluation of the product means that its suitability and safety for use by consumers are checked out. All products made are required by law to be safe to use. This is not a requirement that they are absolutely safe - that is not possible. Nor must they be safe at unbearable costs to industry - that would put innovation at risk. But they are required to be as safe as it is reasonable to expect.

6.2 Future Work

The product can be enhanced based upon requirements. The tool is scalable to other filters and features. The tool can be made available online. Various types of security features can also be implemented for protecting the data when the tool is to used online.

6.3 Conclusion

The tool is quite feasible and confirming to the objectives. We have presented a tool titled as "Graphicizer", build using JAVA technologies. The tool is less constrained and less complex as compared to other products.

Chapter 7

References

1. Java: The Complete Reference, J2SE 5 Edition: Herbert Schildt
2. Java Black Book: The Complete Reference for J2SE
3. http://www.java2s.com/
4. http://homepages.inf.ed.ac.uk/rbf/CVonline/transf.htm

Chapter 8

Appendix

8.1 User Manuals

8.1.1 Introduction

The Graphicizer is an image manipulation tool. The user can open, manipulate & save an image in various formats. The manipulations include operations like blurring, sharpening, bright, contrast, etc.

8.1.2 File menu operations

a. New Image Click on File, select New. Or, click on Open button in the toolbar

b.Opening an image Click on File, select Open. Browse the image file to be opened. Then, click OK.

c. Saving an image Click on File, select Save. Write the full path of the image with file extension in the appearing dialog.

8.1.3 Edit Menu

a. Undo Operation After doing any operation on image, user can undo it by clicking on Edit and selecting Undo.Or click Undo button on the toolbar.

b. Redo Operation After undoing any operation, user can redo it by clicking on Edit and selecting Redo.Or click Redo button on the toolbar.

8.1.4 Image Transformation After opening an image

a. Blurring an image Click on Filter, select Blur.

b. Sharpening an image Click on Filter, select Sharpening.

c. Image Negative Click on Filter, select Image Negative.

d. GrayScale Conversion Click on , select GrayScale Conversion.

e. Brightness Click on , select Brightness.

f. Contrast Click on , select Contrast.

g. Edge Detection Click on , select Edge Detection

8.2 Installation Guide

8.2.1 step 1

Running An Executable JAR From Command Line

You can run it from the command line like this:
java -jar example.jar

Running An Executable JAR From Explorer (Windows)

Within Windows, it is also possible to set up Windows Explorer so that you can double click on the JAR file icon to execute the file (handy for GUI applications). The first thing you must do is set up the correct association with the 'javaw.exe' application that JDK for Windows will have. Click here for an older example with pictures! Open Windows Explorer and select Tools — Folder Options — File Types.

If there is no JAR file type, create it. Give it a description like
jar - Executable Jar File
to ensure it sorts under 'jar'. Create or edit the action called "open" and associate it with the following action:
"C:\Java\jdk1.4.0\bin\javaw.exe" -jar "
Of course, you will replace "C:\Java\jdk1.4.0\bin\javaw.exe" with whatever path is correct on your machine.

IMPORTANT: include the double quotes to take care of names with spaces.

If you are using something other than Windows and you know how to set up an association in your OS, please contact with developers team.

8.3 Technologies Used

- **JDK 5.0 Kit**:
  This is the java environment kit.

- **JCreater**:
  This tool is used as JAVA editor for coding.
- **Latex**:
  Latex is used for documentation.
- **Visual Paradigm tool**:
  This tool is used for creating Class diagrams, Sequence diagrams, UML diagrams.
- **Dia**:
  This tool is used for creating Flow Charts, Data Flow Diagrams and Entity Relationship Diagrams.