CHAPTER 1.

INTRODUCTION

1. AIM OF THE PROJECT

There is need to increase agriculture products in the market as human population is increasing day by day in India. The concern of this project is to use the modern technology in agriculture to increase the product quality and quantity.

A lot of research is going on related to greenhouse automation. But all of these research focus on various parameters in greenhouse such as temperature and humidity. The current image processing practices are done for harvesting of fruits using agricultural robot. There is little concentration on image processing practices for greenhouse crop monitoring.

Therefore, this project focuses on image processing for monitoring greenhouse crop status. The monitoring of ripeness degree of tomatoes in greenhouse is first step towards the crop health status monitoring using android phone for image processing and GSM (Group Special Mobile or Global System for Mobile Communication) technology. For some critical plants such as vegetables and flowers, they need 24 hours attention from human so that the quantities and qualities of the plant can be controlled [1]. Thus, manual supervision plays important role in crop health status monitoring.

With the help of android phone this task can be automated. The human interaction in this case is reduced. In this project, we have taken a tomato fruit as case study. We consider three main stages of ripening for tomatoes. The three stages are unripened (or raw), medium ripened and fully ripened or over ripened.

CHAPTER 2.
LITERATURE SURVEY

2.1. THEORY BEHIND THE PROJECT

The above proposed system turns android based mobile phone into a remote monitoring device that can monitor plants in greenhouse The software does have a few limitations and requirements though.

First of all, mobile phone must have at least 2MP camera and a valid SIM with enough balance to send SMS. The user installs a android application on the phone. Application is programmed in java.

In India the ever-increasing population, losses in handling and processing of crop and the increased expectation of food products a need for the growth of high quality and safety standard.. There is also need of accurate, fast and objective quality determination of food and agricultural products. In conventional method of farming, human labors were required to visit the greenhouse at daily or periodically at specific time and need to check the status of fruit . According to the Sarig, (2005) cost of harvesting by labors is very expensive and time-consuming. It is necessary to reduce the labor effort to monitoring the greenhouse by automation of greenhouse . In agriculture due to recent technology in the field of automation in agricultural yields better productivity, efficiency, cost reduction, and automation . Many technologies have been investigated to be employed in important farming processes for disease detection and product classification

Classification:

1. In-field analysis
2. Post-harvest analysis.

In-field fruit analysis can help detect disease, insects, and a water insufficiency at early stages of growth. In addition, it can also be used for yield prediction and fruit grading and harvesting at appropriate time for negotiating with possible buyers. In post-harvest fruit analysis, there has been some work specific to pineapple. Kaewapichai, Kaewtrakulpong, and Prateepasen [2] employ a technique based on pineapple skincolor for maturity grading. Magnetic resonance imaging (MRI) has also been proposed [3] for textural change measurement due to fruit ripening. We propose a system which is build on infield analysis. the propose system use automated messaging for the ripen fruits in greenhouse with the help of application which uses

mobile phone based on the Android system . This system include Fruit analysis which is important process for provide information about the ripen fruits to the user which are ready to harvest. It replaces the monitoring of status of fruit by human labors and focuses on periodical analysis of fruits by the mobile phone allocated in the greenhouse which capture the images of the plant by using inbuilt camera and perform image processing to collect information about the status of plant such as the percentage of fruits are ready to harvest(or ripened) and send SMS to the user about it .

Tomato is grown worldwide for its edible fruits which has many vitamins and beneficial nutrients. Thousands of cultivars have been selected with different fruit types, and for optimum growth in various growing conditions (Moneruzzaman et al., 2009) [3].Thus we have selected tomato fruit as a case study to implement this project.

2.2 RELATED WORKS

The design requirements for building a computer mediated fruit sorting system vary from fruit to fruit (product to product as well) for which it is designed to process [4]. The capabilities of digital image analysis technology to generate precise descriptive data on pictorial information have contributed to its more widespread and increased use [5]. Image processing technology has been applied to detect cherry tomato in a bunch of cherry tomato plants by recognizing object with different color. The method was integrated in the harvesting robot for cherry tomato cultivated in a greenhouse [6]. The algorithm in [7] is designed with the aim of calculating different weights for features like intensity, color, orientation and edge of the input test image. The weights of different features represent the approximate locations of the fruit within an image. The Detection Efficiency is achieved up to 90% for
different fruit image on tree, captured at different positions. In terms of color characteristic analysis of tomato images and test of image segmentation, RGB and HSI spaces may be implemented to realize dynamic threshold segmentation of color of fruit images in which color of objects and background show noticeable difference [8].

2.2. PROBLEM STATEMENT

The software which we are making is used for automation in agricultural greenhouse.

The aim of our project is to achieve the automation in agricultural greenhouse by developing the software on android based phone that can be used to control the greenhouse remotely in a simple way and avoid the human interventions.
Android phone is fitted in a greenhouse or placed on a robot which will move in a greenhouse. Android phone will capture the images of fruits when timer is started. Then image processing is carried out on a captured image. After that automatically the degree of ripeness of fruits is calculated and user will come to know about the degree of ripeness of fruit automatically through SMS.

2.3. NEED OF THE PROJECT

The final year of engineering comes with its own opportunities and challenges which help the aspiring engineers to tackle complex problems with ease and patience. It makes them apply their knowledge and skills and understand newer and newer concepts. The basic aim for taking this new topic as our final year project was to meet the challenges given by this subject and to apply our skills and knowledge to this complex problem solving.
The aim of our project is to achieve the automation in agricultural greenhouse by developing the software on android based phone that can be used to control the greenhouse remotely in a simple way and avoid the human interventions
The software which we are making is used for automation in agricultural greenhouse. This will help to overcome the problems which are occure due to human interventions. Currently this project is a small scale application. This project, if implemented on large scale in big farms where automation is required to reduce the problems which may occur due to human interventions will help in faster and smoother handling of fruits.

CHAPTER 3.

ANALYSIS AND DESIGN

3.1 ANALYSIS

3.1.1 Process Model:

Software Development Life Cycles is a structured methodology used in the development of software products and packages. This methodology is used from the conception phase through to the delivery and end of life of a final software product. In our software project we use a Waterfall type approach to software development

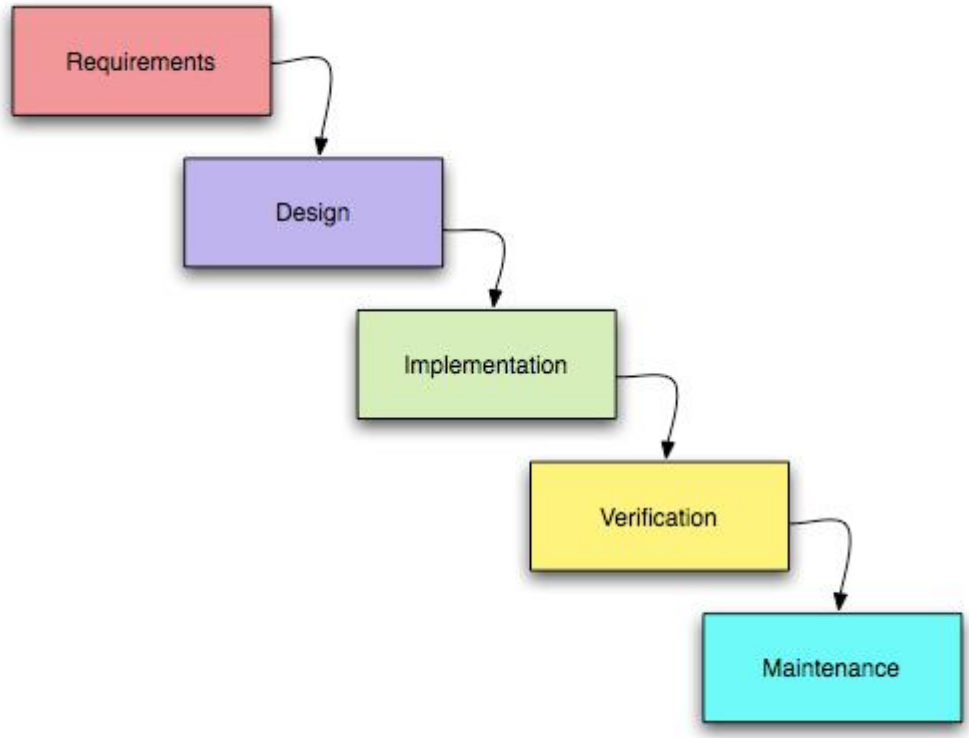


Fig1: Phases of Software Development

As shown in the above diagram of the 'Waterfall Model' First in the software development process, the *requirements phase* outlines the goals of what the program will be capable of doing. Next, the *design phase* covers how the program is going to be created, who will be doing what, etc. The *implementation phase* is where the programmers and other designers start work on the program. After the developers have

a working copy, the *testing and verification* step can begin to help verify the program has no errors. During the testing phase, problems found are fixed, until the program meets the company's quality controls. After the program's development, the *documentation phase* on how to use the program can be completed. Finally, *maintenance and updating* the program must continue for several years after the initial release.

A software development process makes everything easier and reduces the amount of problems encountered.

3.1.2 Feasibility Study:

The feasibility study is a formal proposal for a new system. Before the project is to begin, the project is studied to determine what exactly the user wants depending upon the result of initial investigation.

- **Technical Feasibility:**

Technical feasibility is composed of estimating:

- Hardware cost.
- Software cost.

Hardware cost includes the purchasing of hardware required to develop the project such as a number of system units, monitor, keyboard, mouse and printer.

Software cost includes the cost of purchasing the software required to develop the project, such as operating system (OS), server, front end and back end.

The software required to develop this application is mainly open source softwares and hence freely available on the internet. The hardware is also easily available in the phone market.

- **Operational Feasibility:**

The operational feasibility is composed of estimating the:

- Operational benefits.
- Operational cost.

Operational benefit is in terms of degree to which the new software will be able to solve the current system problems by reducing the difference between the current and desired situation. It means that how much new project is operationally profitable for the

organization such as after implementing and how much cost will save after improving the current operation.

This application can be installed on android based phone like any other applications. Therefore, it can be operated 24x7. But application itself depends on various environmental factors such as light conditions, season of fruit , etc and hardware functionality such as focus of camera, resolution of phone, quality and clarity of camera, balance conditions of GSM card used in the phone etc. But apart from all this limitations overall use of application can be increased if proper care is taken for reducing the effect of resolution. There can be other drawbacks also present such as failure of network, allocation of mobile in greenhouse. .

- **Economic Feasibility:**

From the cost benefit analysis of the hardware and software requirement carried out we can conclude that the system is economically feasible. The cost in developing is negligible when compared to the feature provided by it.

Though this is research base project, there is no such cost associated with project as all the softwares are open source and freely available on the internet. Also the hardware i.e android phone is widely used in modern humans life.

3.1.3 Project Time Line Chart:

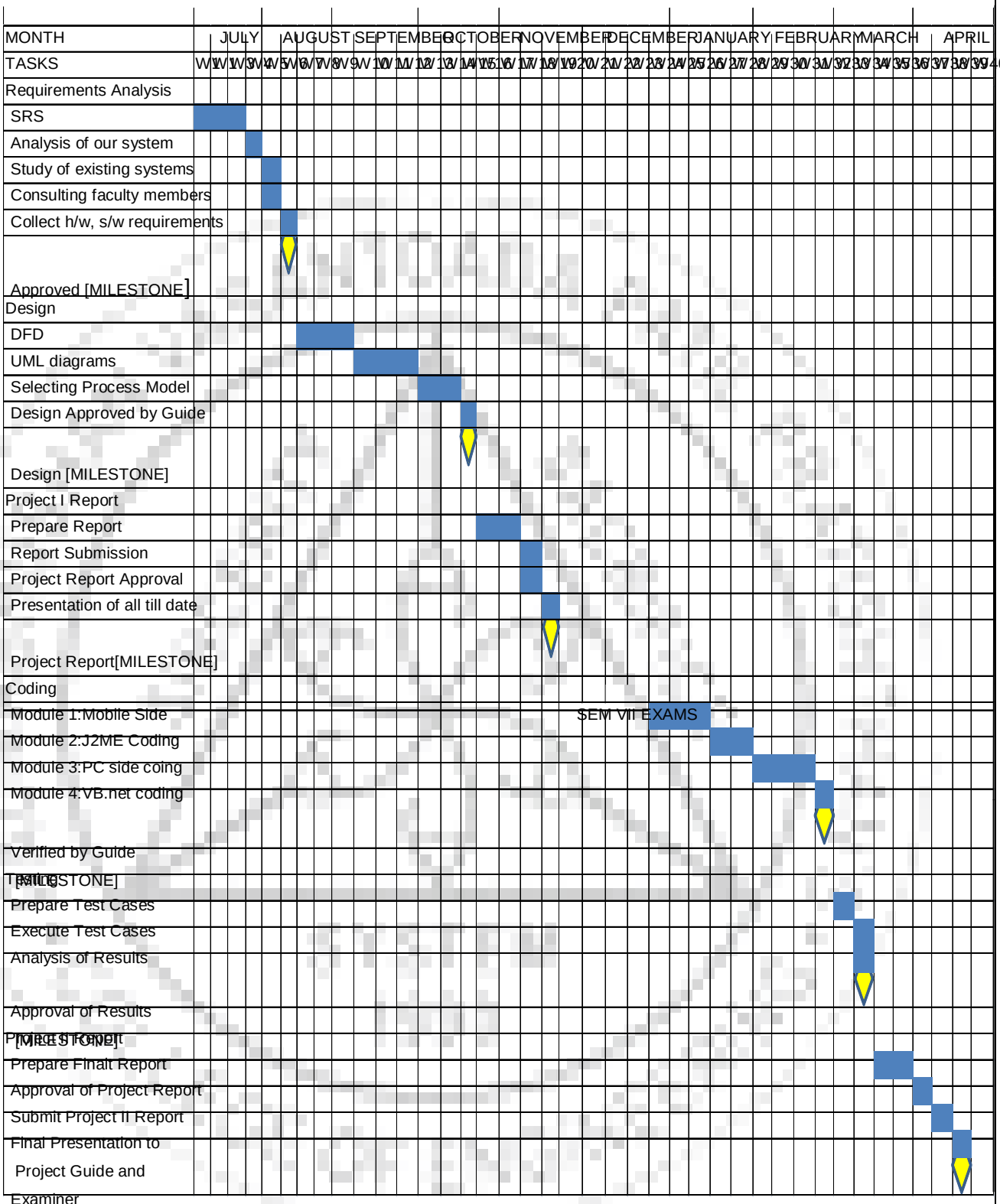

MONTH
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER
JANUARY
FEBRUARY
MARCH
APRIL
TASKS
Requirements Analysis
SRS
Analysis of our system
Study of existing systems
Consulting faculty members
Collect h/w, s/w requirements
Approved [MILESTONE]
Design
DFD
UML diagrams
Selecting Process Model
Design Approved by Guide
Design [MILESTONE]
Project I Report
Prepare Report
Report Submission
Project Report Approval
Presentation of all till date
Project Report[MILESTONE]
Coding
Module 1:Mobile Side
SEM VII EXAMS
Module 2:J2ME Coding
Module 3:PC side coing
Module 4:VB.net coding
Verified by Guide
[MILESTONE]
Testing
Prepare Test Cases
Execute Test Cases
Analysis of Results
Approval of Results
[MILESTONE]
Project II Report
Prepare Finalt Report
Approval of Project Report
Submit Project II Report
Final Presentation to
Project Guide and
Examiner

Table 1: Project Time Line Chart

SYSTEM PLANNING OF GANTT CHART

A Gantt chart is a graphical representation of the duration of tasks against the progression of time. A Gantt chart is useful tool for planning and scheduling projects. It is helpful for monitoring project's progress. The strength of the Gantt chart is its ability to display the status of each activity at a glance.

A Gantt chart is constructed with a horizontal axis representing the total time span of the project, broken down into increments and a vertical axis representing the tasks that make up the project.

Moreover, Gantt chart do not represent the size of the project or the relative size of work elements, therefore the magnitude of a behind schedule condition is easily miscommunicated.

System Maintenance

Types of maintenance

- Corrective Maintenance
- Adaptive Maintenance
- Perfective Maintenance
- Preventive Maintenance

Corrective Maintenance:

Even with the best quality it is likely to have a customer will uncover defects in the software. Corrective Maintenance Changes the software to correct the defects.

Adaptive Maintenance:

The original environment of which the software was developed such as lighting conditions is likely to change. Adaptive Maintenance results in modification to the software to accommodate changes to its external environment.

Perfective Maintenance:

There is need to perform more study for this software application in order to make it perfect for end users to use it. Perfective Maintenance extends the software to its original function requirements.

Preventive Maintenance

Software deteriorates due to change made it in time to time in order to improve it , and because of this Preventive Maintenance also called software reengineering ,and must be conducted to enable the software to serve the needs of its end users. Preventive Maintenance makes changes to computer programs so that they can be easily corrected, adapted and enhanced.

3.2 DESIGN

3.2.1 Data Flow diagrams:

Object oriented analysis and design is a software engineering approach that models a system as a group of interacting objects.

Each object represents some entity of interest in the system being modeled, and is characterized by its class, its state and its behavior.

Various models can be created to show the static structure, dynamic behavior and run-time deployment f this objects.

There are no of different notation for representing these models ,such as Unified Modeling Language.

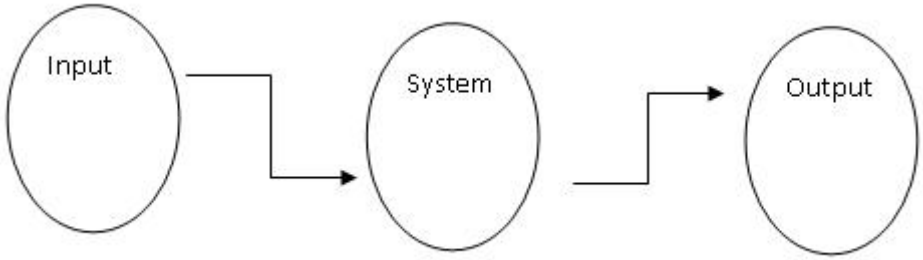
We implemented this record in our model because:-

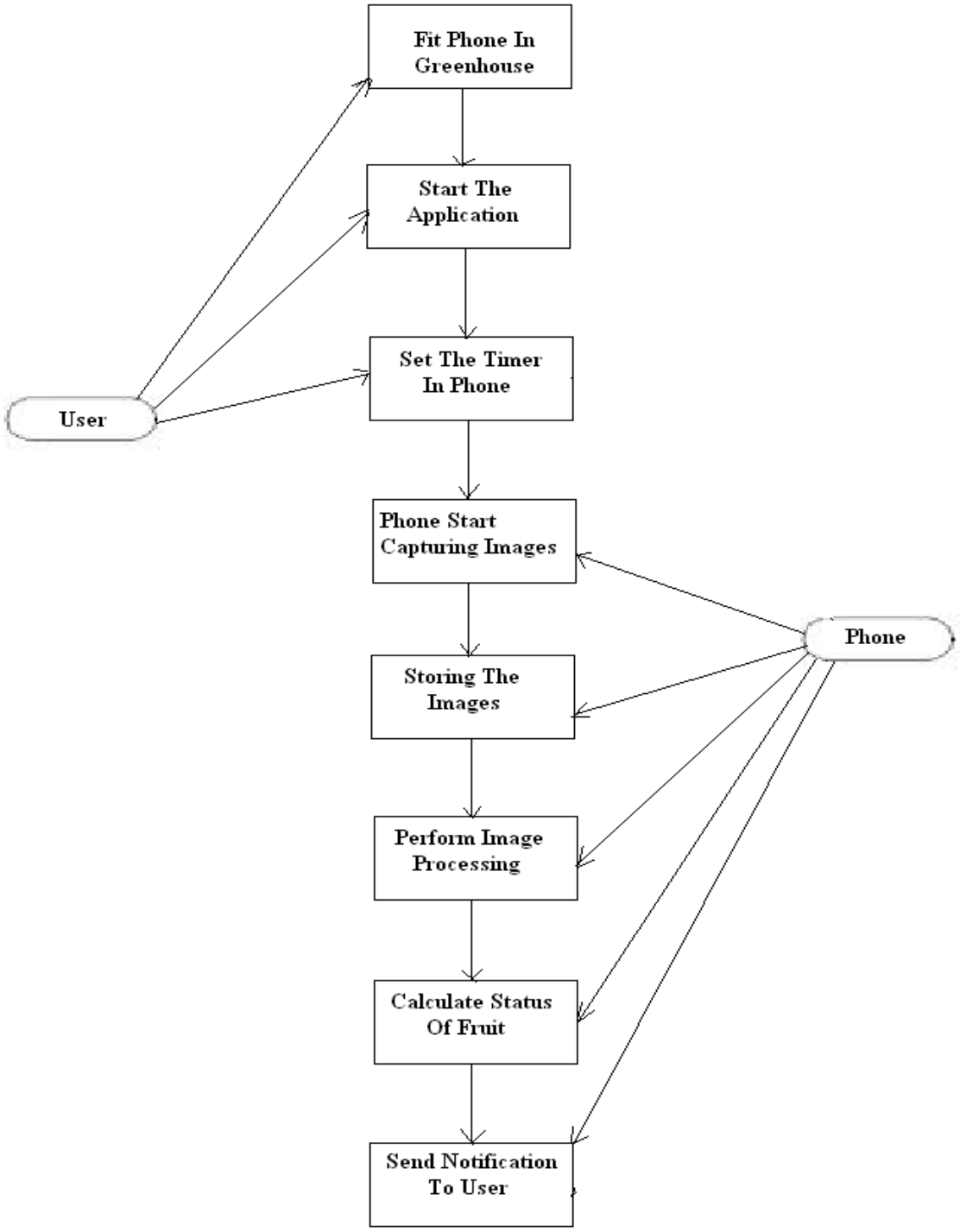
I. UML is Language for visualizing, specifying, constructing and documenting the artifacts of a software intensive system.

II. UML language provides vocabulary rules for combining words in that vocabulary for the purpose of communication.

III. A modeling language is language whose vocabulary and rules focuses on the conceptual representation of a system language for a software blueprints

DFD LEVEL 0:


Input
System
Output


Fit Phone In Greenhouse
Start The Application
Set The Timer In Phone
User
Phone Start Capturing Images
Phone
Storing The Images
Perform Image Processing
Calculate Status Of Fruit
Send Notification To User

DFD LEVEL 1:

Fig. 2: Data Flow Diagram

3.2.2 UML diagrams:

1) Use Case Diagram:

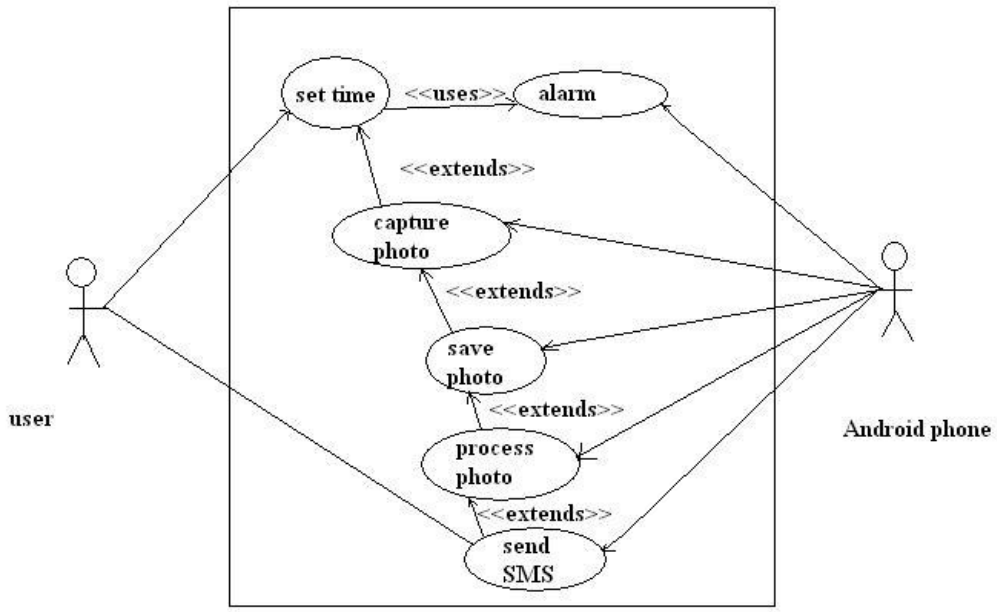


Fig.3: Use Case Diagram

2) Activity Diagram

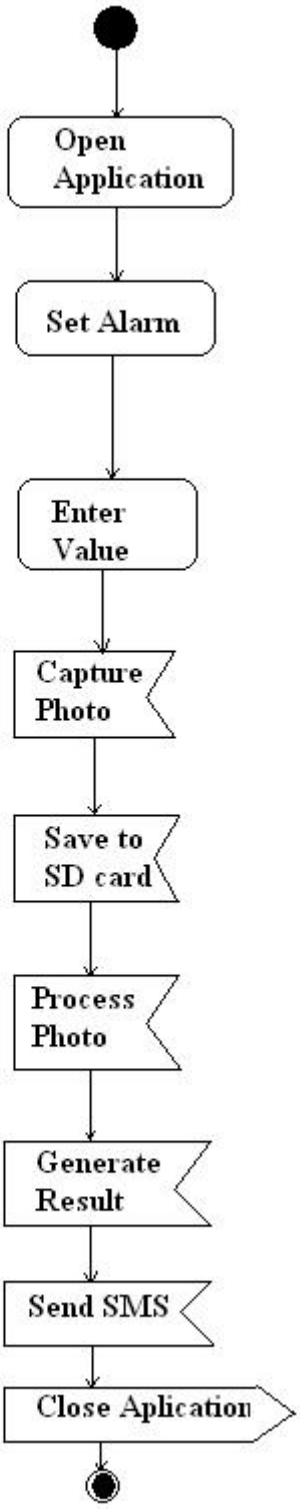


Fig 4: Activity Diagram

3) Component Diagram:

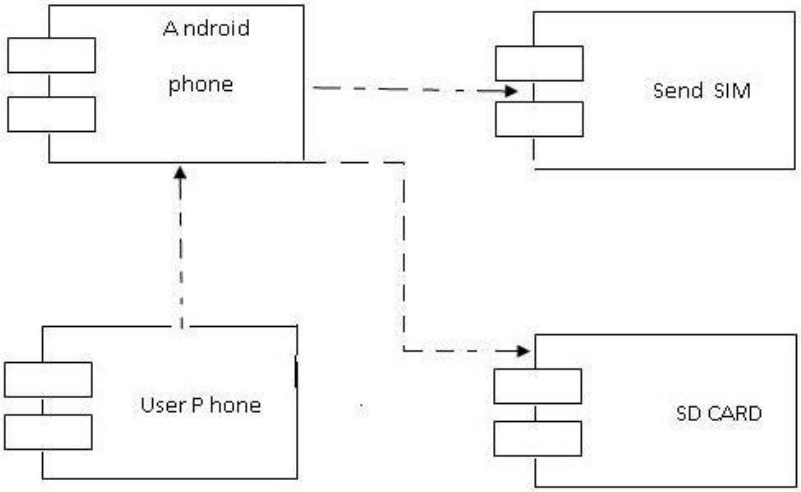


Fig 5: Component Diagram

4) Sequence Diagram:

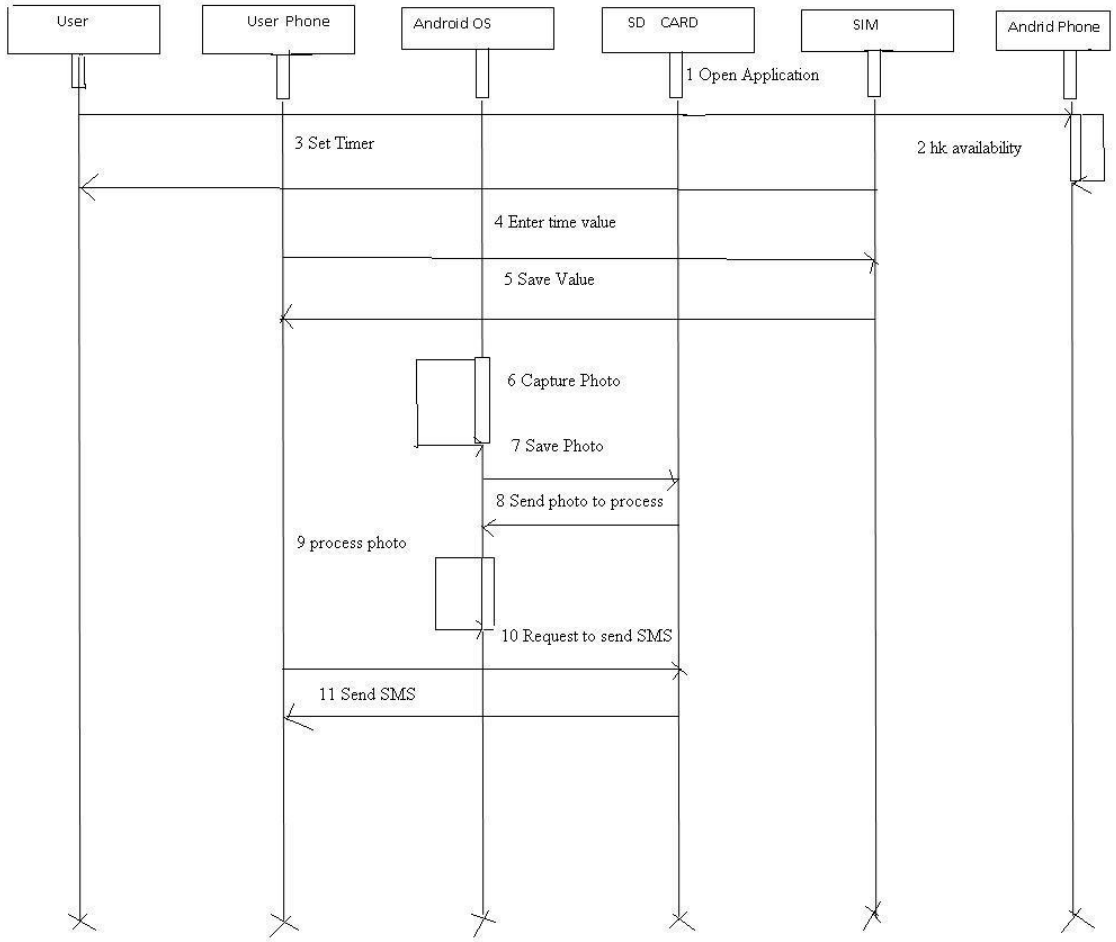
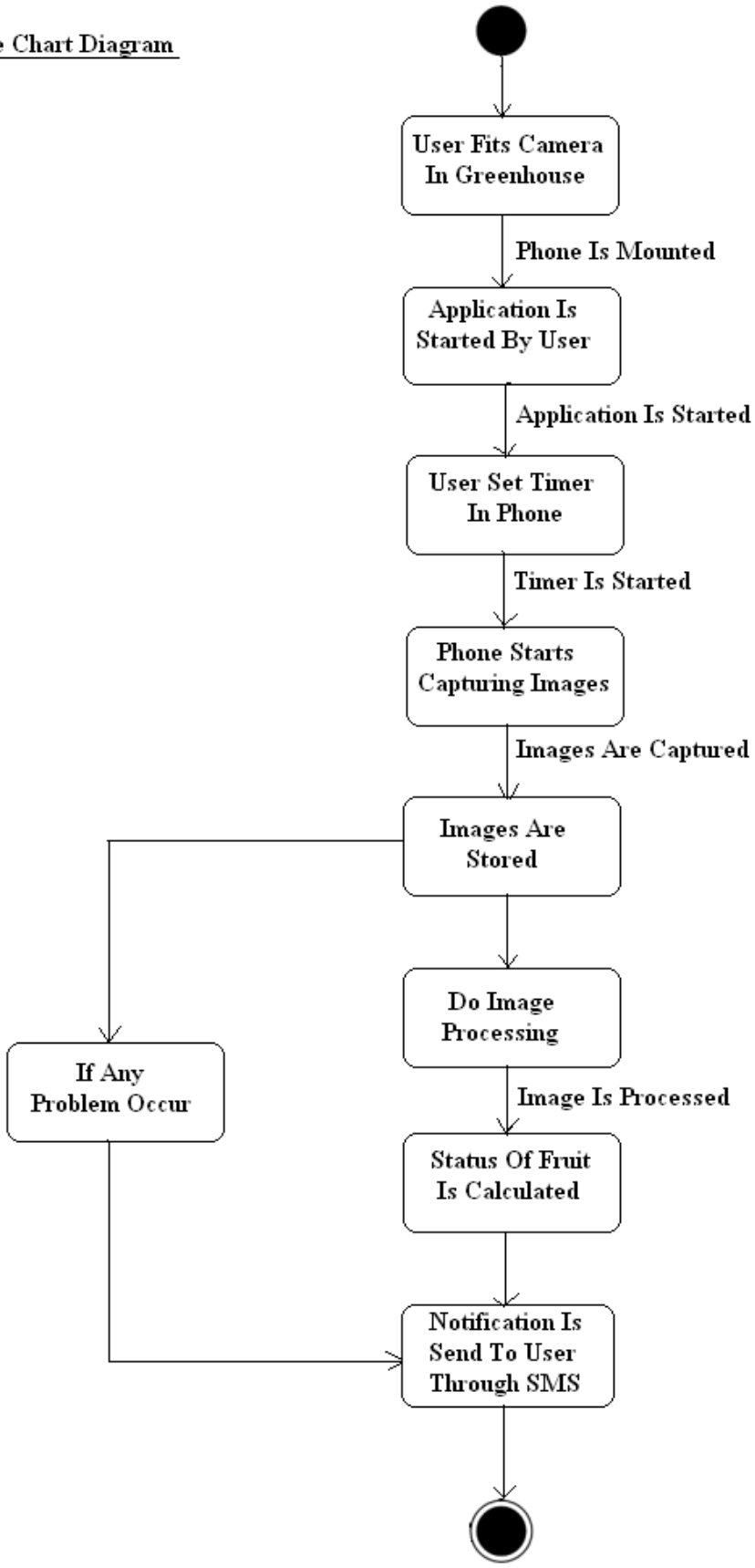


Fig 6: Sequence Diagram

5) State Chart Diagram:

Fig 7: State chart Diagram

State Chart Diagram



CHAPTER 4.
REQUIREMENTS

4.1. HARDWARE REQUIREMENTS:

The hardware used for the development of the project is:

PROCESSOR : PENTIUM IV 2.40GHz OR HIGHER

RAM : 0.99 RAM

HARD DISK : 160 GB

MOBILE DEVICE : ANDROID BASED PHONE WITH AT LEAST 2MP CAMERA AND SD CARD

4.2. SOFTWARE REQUIREMNETS: -

The software used for the development of the project is:

OPERATING SYSTEM : Windows 7, Windows Vista, Windows XP with or later operating system (32 or 64 bit)

ECLIPSE IDE : Eclipse IDE for Java Developers

JAVA : JDK 6

ADT Plug-ins and Android SDK

4.3. TECHNOLOGY USED:

4.3.1. Android Operation System:

Android is an operating system based on Linux with a Java programming interface.

The Android Software Development Kit (Android SDK) provides all necessary tools to develop Android applications. This includes a compiler, debugger and a device emulator, as well as its own virtual machine to run Android programs.

Android is currently primarily developed by Google.

Android allows background processing, provides a rich user interface library, supports 2-D and 3-D graphics using the OpenGL libraries, access to the file system and provides an embedded SQLite database.

Android applications consist of different components and can re-use components of other applications, if these applications declare their components as available. This leads to the concept of a task in Android; an application can re-use other Android components to archive a task.

For example you can write an application which integrates a map component and a camera component to archive a certain task.

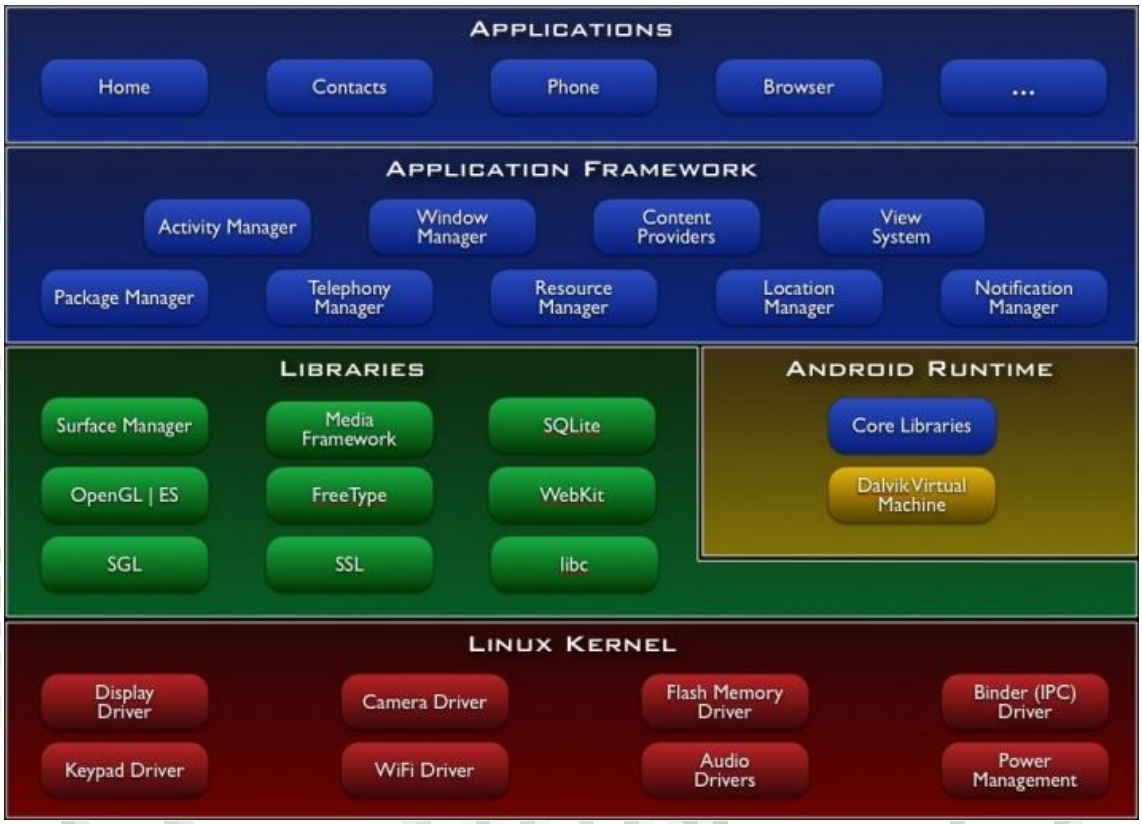


Figure 8: Android software stack

4.3.2. Android components:

The following gives a short overview of the most important Android components.

a. Activity

Activity represents the presentation layer of an Android application. A simplified (and slightly incorrect) description is that an Activity is a screen. This is slightly incorrect as Activities can be displayed as Dialogs or can be transparent. An Android application can have several Activities.

b. Views and ViewGroups

Views are user interface widgets, e.g. buttons or text fields. The base class for all Views isandroid.view.View. Views often have attributes which can be used to change their appearance and behavior.

A ViewGroup is responsible for arranging other Views e.g. a ViewGroup is a layout manager. The base class for a layout manager is android.view.ViewGroups. ViewGroup also extends View.ViewGroups can be nestled to create complex layouts. You should not nestle ViewGroups too deeply as this has a negative impact on the performance.

c. Intents

Intents are asynchronous messages which allow the application to request functionality from other components of the Android system, e.g. from Services or Activities. An application can call a component directly (explicit Intent) or ask the Android system to evaluate registered components for a certain Intent (implicit Intents). For example the application could implement sharing of data via anIntent and all components which allow sharing of data would be available for the user to select. Applications register themselves to an Intent via an IntentFilter.

Intents allow to combine loosely coupled components to perform certain tasks.

d. Services

Services perform background tasks without providing a user interface. They can notify the user via the notification framework in Android.

e. Content Provider

ContentProvider provides a structured interface to application data. Via a ContentProvider your application can share data with other applications. Android contains an SQLite database which is frequently used in conjunction with a ContentProvider to persist the data of the ContentProvider.

f. Broadcast Receiver

BroadcastReceiver can be registered to receive system messages and Intents. ABroadcastReceiver will get notified by the Android system, if the specified situation happens. For example a BroadcastReceiver could get called once the Android system completed the boot process or if a phone call is received.

g. (HomeScreen) Widgets

Widgets are interactive components which are primarily used on the Android homescreen. They typically display some kind of data and allow the user to perform actions via them. For example a Widget could display a short summary of new emails and if the user selects an email, it could start the email application with the selected email.

h. Other

Android provide many more components but the list above describes the most important ones. Other Android components are "Live Folders" and "Live Wallpapers". Live Folders display data on the homescreen without launching the corresponding application.

4.3.3. Android Development Tools

Google provides the Android Development Tools (ADT) to develop Android applications with Eclipse. ADT is a set of components (plug-ins) which extend the Eclipse IDE with Android development capabilities.

ADT contains all required functionalities to create, compile, debug and deploy Android applications from the Eclipse IDE and from the command line. Other IDE's, e.g. IntellJ, are also reusing components of ADT.

ADT also provides an Android device emulator, so that Android applications can be tested without a real Android phone.

a. Dalvik Virtual Machine

The Android system uses a special virtual machine, i.e. the Dalvik Virtual Machine to run Java based applications. Dalvik uses an own bytecode format which is different from Java bytecode.

Therefore you cannot directly run Java class files on Android, they need to get converted in the Dalvik bytecode format.

4.3.4. How to develop Android Applications

Android applications are primarily written in the Java programming language. The Java source files are converted to Java class files by the Java compiler.

Android provides a tool called "dx"" which converts Java class files into a dex (Dalvik Executable) file. All class files of one application are placed in one compressed .dex file. During this conversion process redundant information in the class files are optimized in the .dex file. For example if the same String is found in different class files, the .dex file contains only once reference of this String.

These dex files are therefore much smaller in size than the corresponding class files.

The .dex file and the resources of an Android project, e.g. the images and XML files, are packed into an.apk (Android Package) file. The program aapt (Android Asset Packaging Tool) performs this packaging.

The resulting .apk file contains all necessary data to run the Android application and can be deployed to an Android device via the "adb" tool.

The Android Development Tools (ADT) allows that all these steps are performed transparently to the user; either within Eclipse or via the command line.

If you use the ADT tooling you press a button or run a script and the whole Android application (.apk file) will be created and deployed.

4.3.5. Eclipse :

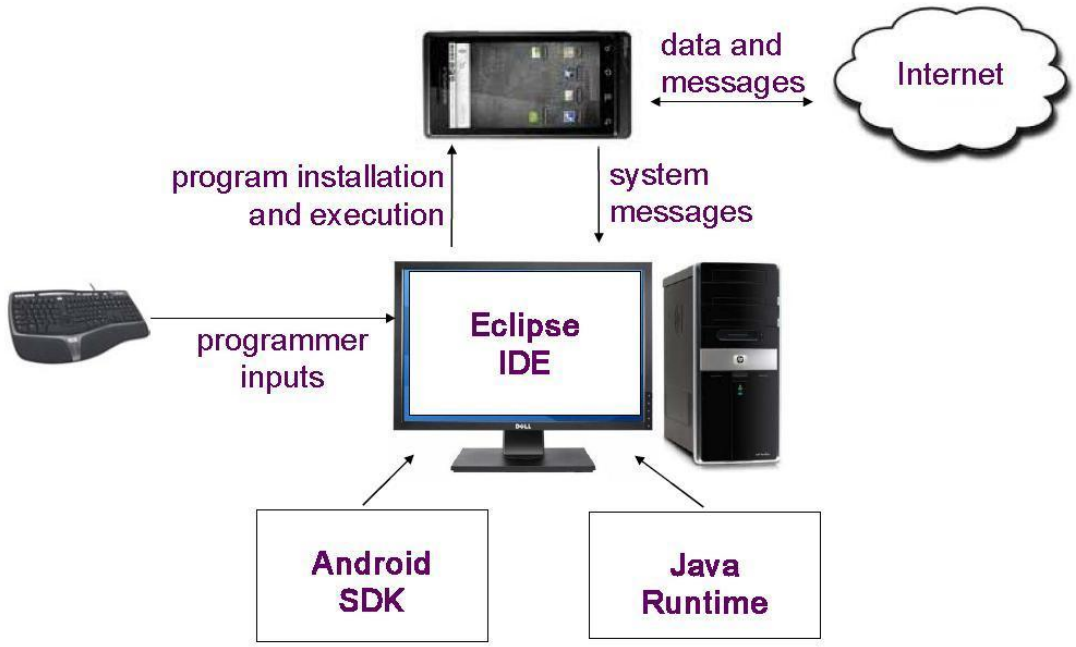


Figure 9: Android programming with Eclipse

Eclipse is created by an Open Source community and is used in several different areas, e.g. as a development environment for Java or Android applications. Eclipse roots go back to 2001.

Most people know Eclipse as an integrated development environment (IDE) for Java. Today it is the leading development environment for Java with a marketshare of approx. 65%.

The Eclipse project is governed by the Eclipse Foundation. The Eclipse Foundation is a non-profit, member supported corporation that hosts the Eclipse projects and helps to cultivate both an open source community and an ecosystem of complementary products and services.

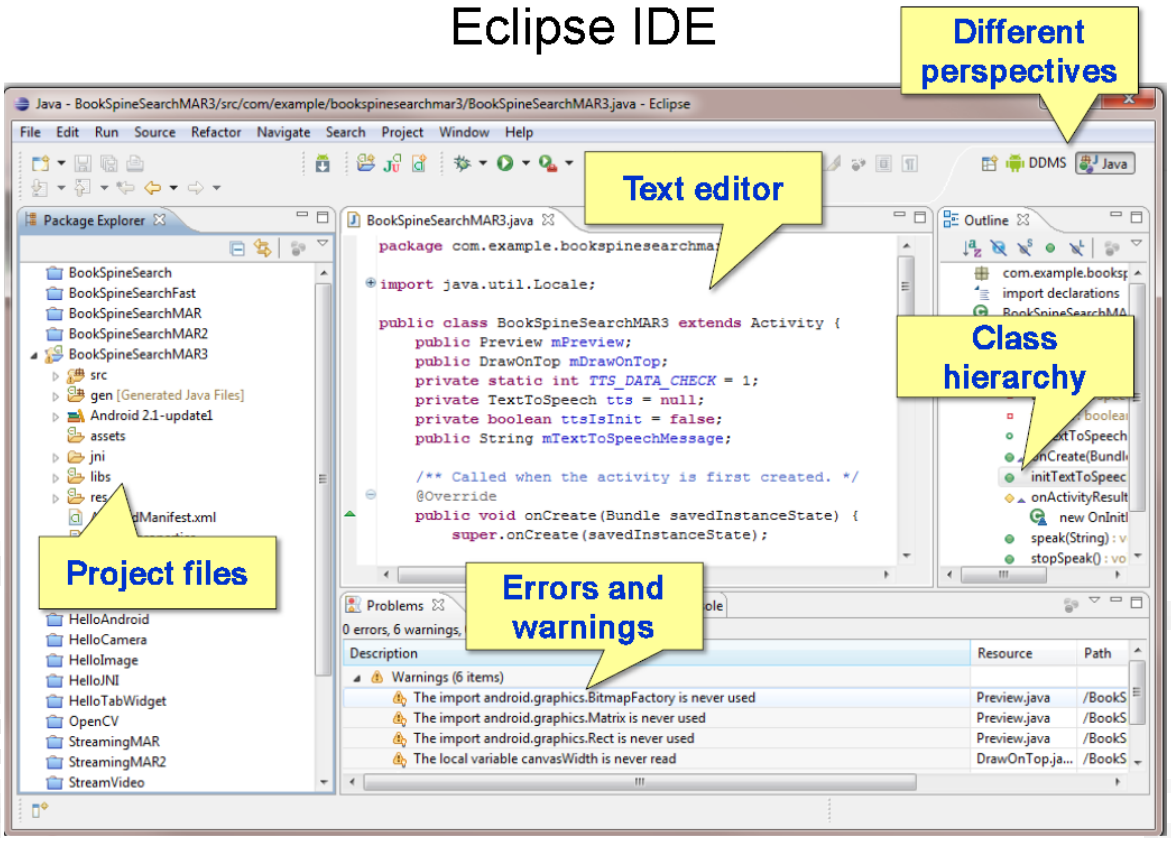


Figure 10: Eclipse IDE

Eclipse can be extended with additional functionalities. Several Open Source projects and companies have extended Eclipse with additional components. It is also possible to use Eclipse as a basis for creating general purpose applications (Eclipse RCP).

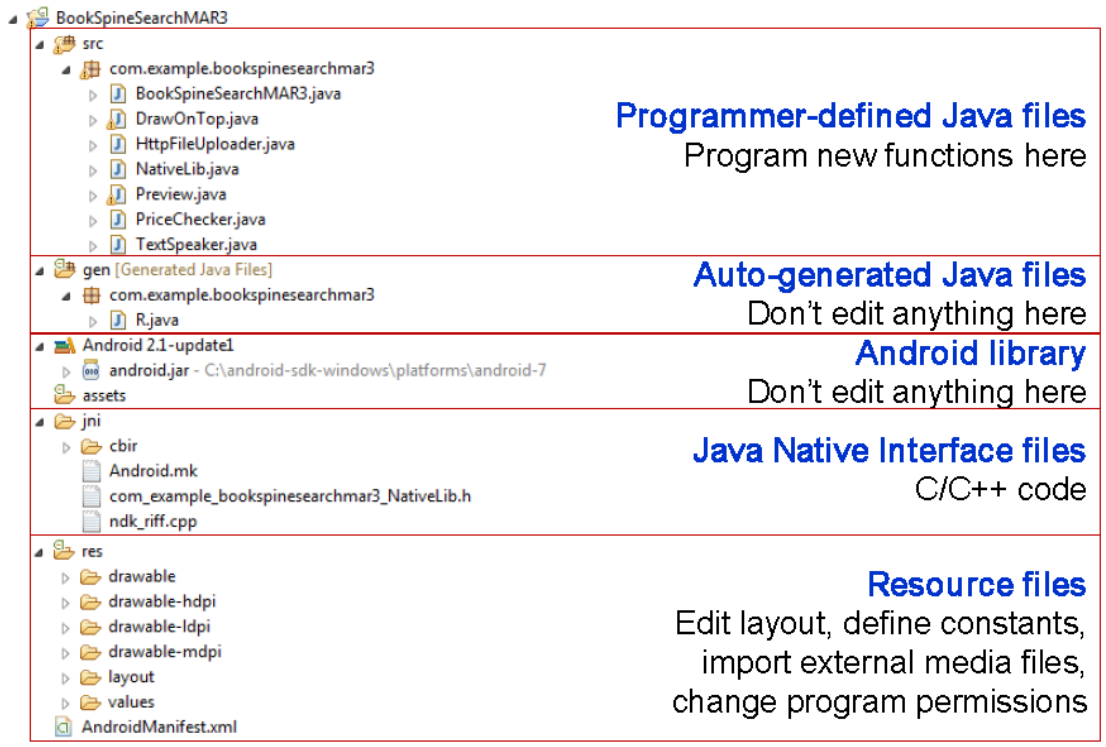


Figure 11 : Structure of an Android Project in Eclipse

CHAPTER 5.
IMPLEMENTATION PROCEDURE

Implementation is the stage of the project where the theoretical design is turned into a working system at this stage and the major impact on the existing practices shifts to the user department.

Thus it can be considered as a most crucial stage in achieving a successful new system and in giving the user's confident that the new system will work properly and be effective.

Fig.12: Connection of android phone with computer

5.1 INSTALLATION GUIDE:

STEP 1: Install JAVA

STEP 2: Install Eclipse IDE for Java Developers.

http://www.eclipse.org/downloads

STEP 3: Download SDK starter package and install it.

STEP 4: Install ADT plug-ins for Eclipse

STEP 5: Add platform(API level 8) and other packages

STEP 6: Linking Your Phone to Your Computer

Creating the Software Development Environment

We are using the Google Android SDK, the Java JDK, and the Eclipse IDE to design, implement, and debug Android-compatible programs in this class.

I. Downloading and Installing Java JDK

1. Download the installer file for the Java SE 7 JDK for Windows (either 32 bit or 64 bit depending on your operating system) from this website:

http://www.oracle.com/technetwork/java/javase/downloads/index.html

2. Run the downloaded installer file.

3. Choose the installation folder for the JDK, for example:

C:\\Program Files\\Java\\jdk1.7.0_03

II. Downloading and Installing Eclipse

1. Download "Eclipse IDE for Java Developers (Indigo)" from this website:

http://www.eclipse.org/downloads/

2. Unzip the downloaded file to a convenient location on your hard disk, for example:

C:\\eclipse

3. Open "eclipse.exe" found in the folder where you unzipped the contents. You may want

to create a shortcut in the taskbar for easy launching in the future.

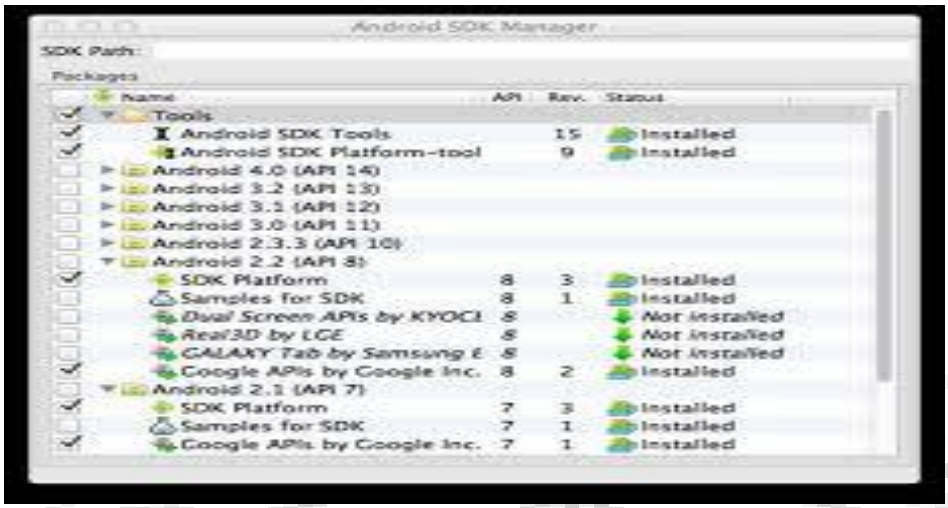
4. When asked to choose a default workspace, pick a folder that is easy to remember and access, for example:

C:\\Users\\Your_Name\\workspace

5. Verify that Eclipse starts properly and an IDE window like in Figure 3 is shown

III. Downloading and Installing Android SDK

Fig. 13 Android SDK Manager



1. Download the Windows installer file installer_r16-windows.exe for the Google Android SDK from this website:

http://developer.android.com/sdk/index.html

2. Run the downloaded installer file.

3. Choose the installation folder for the Android SDK, for example:

C:\\Program Files (x86)\\Android\\android-sdk

4. Add the location of the “tools” and “platform-tools” subfolders for the Android SDK to

your system PATH. For help on editing the PATH, please follow the tips here:

http://www.windowsitpro.com/article/john-savills-windows-faqs/how-cani-add-a-new-folder-to-my-system-path-.aspx3

5. After the installation files are copied, check “Start SDK Manager” and click “Finish”.

6. In the Android SDK Manager that pops up, check at least the following boxes under “Packages”:

Tools, Android 3.0, Android 2.3, Android 2.2, Android 2.1, Android 1.6,

Extras.

7. Click “Install -- packages”, choose “Accept All”, and click “Install”. The selected packages will now be downloaded and copied to your Android SDK installation folder.

8. During the download, if you are asked for Motorola or HTC developer account information, you can register for free accounts at:

http://developer.motorola.com

http://htcdev.com

IV. Install the ADT plugin for Eclipse.

a. Open Eclipse.

b. From top menubar, choose Help > Install New Software.

c. In the Available Software dialog, click Add …

d. In the Add Site dialog, enter "Android Plugin" in the "Name" field and enter the following URL in the "Address" field and click OK:

https://dl-ssl.google.com/android/eclipse/

e. In the Available Software dialog, you should see "Developer Tools" listed. Select the checkbox next to "Developer Tools" and click Next.

f. In the Install Details dialog, you should see "Android DDMS", "Android Development Tools", and possibly other tools listed. Click Next, accept all license agreements, and click Finish.

g. Restart Eclipse.

V. Link Eclipse to the installed Android SDK.

1.a. In Eclipse, select Window > Preferences. A window like that in Figure 2 should pop up.

b. Select Android from the left panel.

c. For the "SDK Location", click Browse and find where you installed the Android SDK.4

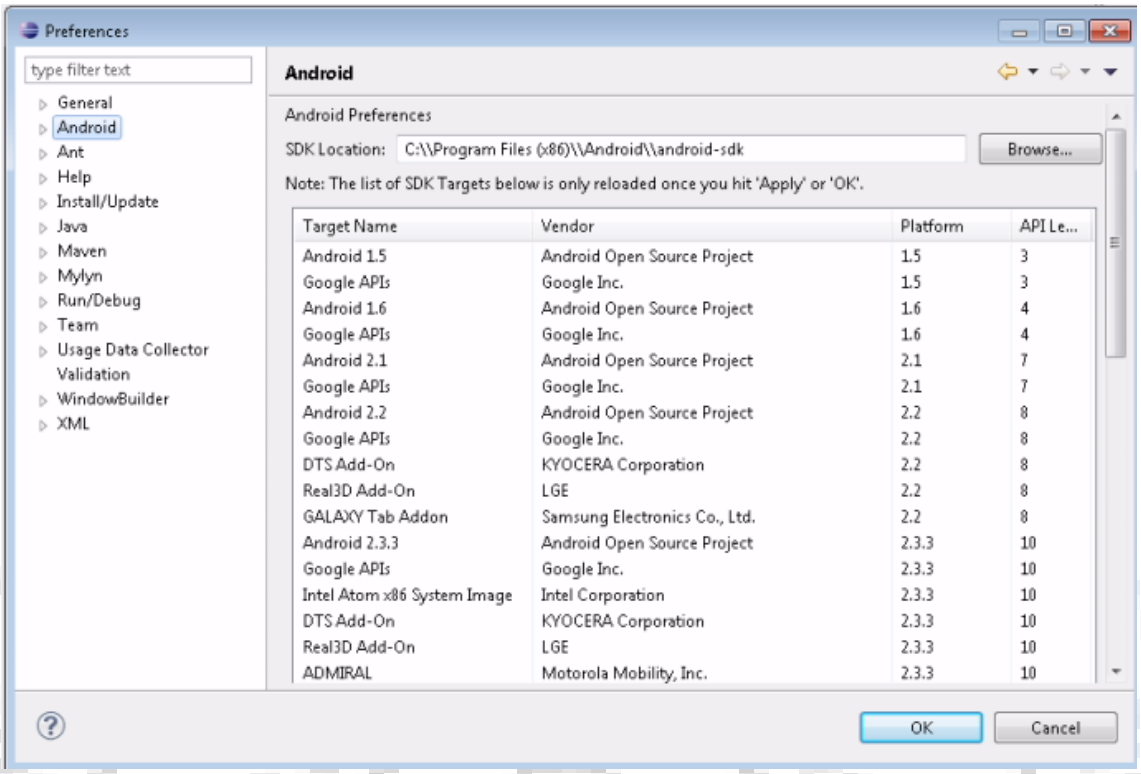


Figure 14. Android preferences panel in Eclipse.

d. Click Apply and then click OK.

e. Restart Eclipse.

2. In Eclipse, select Window > Android SDK Manager and verify that the packages selected in Step 6 appear as “Installed”. If any packages were not installed before, you can also install them at this point.

3. If you encountered problems in this section, please take a look at the tips on these sites:

http://developer.android.com/sdk/index.html

http://developer.android.com/sdk/installing.html

http://developer.android.com/sdk/eclipse-adt.html

VI. Linking Your Phone to Your Computer

1. Connect your phone to your computer via USB. Turn on your phone.

2. Go to the home screen.

3. Select Settings > Applications > Development and then enable USB debugging.

4. After you have downloaded updates for the Android SDK in Eclipse above, the USB

driver should have been included. Install the USB driver on your computer, following the tips on these pages:5

http://developer.android.com/guide/developing/device.html#setting-up

http://developer.android.com/sdk/win-usb.html

5.2 SYSTEM IMPLEMENTATION

The Following are the prominent features of the above discussed system…

- Using android based mobile as remote monitoring device for greenhouse plants,
- Use of GSM and SIM technology to send SMS to user regarding plant status,
- Up to 50cm range,
- Automatic working of android based mobile device, once time is set.

The major tasks performed in this system are:

- Setting the alarm to capture photo at specific time of the day.
- Automatic saving and processing of images captured by phone camera.
- Generating the appropriate results from processing of images.
- Sending the SMS to the user about the result.

I. CODE FOR ALARM SERVICE TO CAPTURE PHOTO AUTOMATICALLY:

```
package com.agribot.automaticcapture;

import android.app.Service;
import android.content.Intent;
import android.hardware.Camera;
import android.os.IBinder;
import android.view.LayoutInflater;
import android.widget.Toast;

public class MyAlarmService extends Service {

	Camera camera;
	Preview preview;
	LayoutInflater layoutInflater;


	@Override
	public void onCreate() {
```

```
	//	Toast.makeText(this, "MyAlarmService.onCreate()",
Toast.LENGTH_LONG).show();
	}

	@Override
	public IBinder onBind(Intent intent) {
		// TODO Auto-generated method stub
		Toast.makeText(this, "MyAlarmService.onBind()",
Toast.LENGTH_LONG).show();
		return null;
	}

	@Override
	public void onDestroy() {
		// TODO Auto-generated method stub
		super.onDestroy();
		Toast.makeText(this, "MyAlarmService.onDestroy()",
Toast.LENGTH_LONG).show();
	}

	@Override
	public void onStart(Intent intent, int startId) {

		Toast.makeText(this, "MyAlarmService.onStart()",
Toast.LENGTH_SHORT).show();
		Intent intent1 = new Intent(getBaseContext(),
CapturePhoto.class);
		intent1.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
		getApplicationContext().startActivity(intent1);
		super.onStart(intent, startId);

	}	@Override
	public boolean onUnbind(Intent intent) {
		// TODO Auto-generated method stub
		return super.onUnbind(intent);
	}


}
```

II. CODE FOR CAPTURING PHOTO:

```
package com.agribot.automaticcapture;

import android.app.Activity;
import android.hardware.Camera;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.FrameLayout;

import com.agribot.dashboard.R;

public class CapturePhoto extends Activity{
	private static final String TAG = "CameraDemo";
	Camera camera;
	Preview preview;
	Button buttonClick;
```

```
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.camerapreview);

		preview = new Preview(this);
		((FrameLayout)
findViewById(R.id.preview)).addView(preview);
			Log.d(TAG, "onCreate'd");
	}


}
```

III. CODE FOR STORING IMAGE ON THE SD CARD:

```
package com.agribot.automaticcapture;

import android.app.Activity;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import com.agribot.dashboard.R;
import com.agrobot.capturephoto.AgribotCapturePhoto;

public class AfterCapture extends Activity {


	ImageView imageView;
	ProgressBar progressBar;
	TextView textView;


	@Override
	protected void onCreate(Bundle savedInstanceState) {
		// TODO Auto-generated method stub
		super.onCreate(savedInstanceState);
		setContentView(R.layout.afterautomaticcapture);
		imageView=(ImageView)
findViewById(R.id.afterimagecapture);
		progressBar=(ProgressBar) findViewById(R.id.progressBar1);
		textView=(TextView)
findViewById(R.id.afterimagecapturetext);
		imageView.setVisibility(View.INVISIBLE);
		textView.setVisibility(View.INVISIBLE);
		progressBar.setVisibility(View.VISIBLE);

		System.out.println(Preview.phototakenname);
```

```
            Bitmap bitmap =
BitmapFactory.decodeFile("/sdcard/"+Preview.phototakenname+".jpg");
            Bitmap newphoto = AgribotCapturePhoto.doGreyscale(bitmap);
            Bitmap newphoto2 = AgribotCapturePhoto.binarize(newphoto,
bitmap);
            String a = "";
            a = AgribotCapturePhoto.findcolor(newphoto2);

            /*imageView.setLayoutParams(new LinearLayout.LayoutParams(
                        LinearLayout.LayoutParams.WRAP_CONTENT,
                        LinearLayout.LayoutParams.WRAP_CONTENT));

            imageView.getLayoutParams().height = 300;
            imageView.getLayoutParams().width = 300;
            imageView.setImageBitmap(newphoto2);*/
            Log.d("", ""+a);
             Toast.makeText(this, a, Toast.LENGTH_LONG).show();
            textView.setText(a);
            imageView.setImageBitmap(newphoto2);
            progressBar.setVisibility(View.GONE);
            imageView.setVisibility(View.VISIBLE);
            textView.setVisibility(View.VISIBLE);

                  }


}
```

IV. CODE FOR PROCESSING OF IMAGE AND SENDING SMS TO USER:

```
package com.agrobot.capturephoto;

import android.app.Activity;
import android.app.PendingIntent;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.Color;
import android.os.Bundle;
import android.telephony.SmsManager;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;

import com.agribot.dashboard.R;

public class AgribotCapturePhoto extends Activity {

      Button button;
      ImageView imageView;
      int requestcode = 114;
      Bitmap photo, newphoto, newphoto2, newphoto3;
      TextView textView;

       public static Bitmap doGreyscale(Bitmap src) {
            // constant factors
```

```
            final double GS_RED = 0.299;
            final double GS_GREEN = 0.587;
            final double GS_BLUE = 0.114;
            // create output bitmap
            Bitmap bmOut = Bitmap.createBitmap(src.getWidth(),
src.getHeight(),
                        src.getConfig());
            // pixel information
            int A, R, G, B;
            int pixel;

            // get image size
            int width = src.getWidth();
            int height = src.getHeight();

            // scan through every single pixel
            for (int x = 0; x < width; ++x) {
                  for (int y = 0; y < height; ++y) {
                        // get one pixel color
                        pixel = src.getPixel(x, y);
                        // retrieve color of all channels
                        A = Color.alpha(pixel);
                        R = Color.red(pixel);
                        G = Color.green(pixel);
                        B = Color.blue(pixel);
                        // take conversion up to one single value
                        R = G = B = (int) (GS_RED * R + GS_GREEN * G +
GS_BLUE * B);
                        // set new pixel color to output bitmap
                        bmOut.setPixel(x, y, Color.argb(A, R, G, B));
                  }
            }

            // return final image
            return bmOut;
      }

      public static Bitmap binarize(Bitmap original, Bitmap
original2) {

            int red;
            int newPixel, pixel;

            int threshold = otsuTreshold(original);
            System.out.println(">>>>>>>>>>>>>>      " + threshold);
            Bitmap binarized =
Bitmap.createBitmap(original.getWidth(),
                        original.getHeight(), original.getConfig());
            int[] pix = new int[original.getWidth() *
original.getHeight()];

            for (int i = 0; i < original.getWidth(); i++) {
                  for (int j = 0; j < original.getHeight(); j++) {

                        float[] hsv = new float[3];

                        pixel = original.getPixel(i, j); // Get pixels
                        int index = i * original.getWidth() + j;
                        int A = Color.alpha(pixel);
                        int R = Color.red(pixel);
```

```
                        Color.colorToHSV(original2.getPixel(i, j),
hsv);

                        float H = hsv[0];
                        float S = hsv[1];
                        float V = hsv[2];

                        if (H > threshold) {
                              newPixel = 255;
                              newPixel = colorToRGB(0xff, newPixel,
newPixel, newPixel);
                              binarized.setPixel(i, j, newPixel);
                        } else {

                              binarized.setPixel(i, j,
Color.HSVToColor(hsv));

                        }
                        // newPixel = colorToRGB(0xff, newPixel,
newPixel, newPixel);

                  }
            }

            return binarized;
      }

       public static int colorToRGB(int alpha, int red, int green, int
blue) {

            int newPixel = 0;
            newPixel += alpha;
            newPixel = newPixel << 8;
            newPixel += red;
            newPixel = newPixel << 8;
            newPixel += green;
            newPixel = newPixel << 8;
            newPixel += blue;

            return newPixel;

      }

       public static int[] imageHistogram(Bitmap input) {

            int[] histogram = new int[256];
            int[] pix = new int[input.getWidth() * input.getHeight()];
            int pixel;
            for (int i = 0; i < histogram.length; i++)
                  histogram[i] = 0;

            for (int i = 0; i < input.getWidth(); i++) {
                  for (int j = 0; j < input.getHeight(); j++) {

                        pixel = input.getPixel(i, j); // Get pixels

                        int index = i * input.getWidth() + j;
                        int R = Color.red(pixel);
                        // bitwise shifting

                        histogram[R]++;
```

```
                }
            }

            return histogram;

        }

        public static int otsuTreshold(Bitmap original) {

            int[] histogram = imageHistogram(original);
            int total = original.getHeight() * original.getWidth();

            float sum = 0;
            for (int i = 0; i < 256; i++)
                sum += i * histogram[i];

            float sumB = 0;
            int wB = 0;
            int wF = 0;

            float varMax = 0;
            int threshold = 0;

            for (int i = 0; i < 256; i++) {
                wB += histogram[i];
                if (wB == 0)
                    continue;
                wF = total - wB;

                if (wF == 0)
                    break;

                sumB += (float) (i * histogram[i]);
                float mB = sumB / wB;
                float mF = (sum - sumB) / wF;

                float varBetween = (float) wB * (float) wF * (mB -
mF) * (mB - mF);

                if (varBetween > varMax) {
                    varMax = varBetween;
                    threshold = i;
                }
            }

            return threshold;

        }

        public static String findcolor(Bitmap newphoto4) {

            //
Toast.makeText(this,""+newphoto4.getHeight()*newphoto4.getWidth(),Toas
t.LENGTH_SHORT);

            int[] pix = new int[newphoto4.getWidth() *
newphoto4.getHeight()];
            int pixel, unripe = 0, medripe = 0, ripe = 0, total = 0;
            for (int i = 0; i < newphoto4.getWidth(); i++) {
                for (int j = 0; j < newphoto4.getHeight(); j++) {
```

```
                        float[] hsv = new float[3];

                        pixel = newphoto4.getPixel(i, j); // Get
pixels
                        Color.colorToHSV(newphoto4.getPixel(i, j),
hsv);


                        float H = hsv[0];
                        float S = hsv[1];
                        float V = hsv[2];
                        int R = Color.red(pixel);
                        int G = Color.green(pixel);
                        int B = Color.blue(pixel);

                        //System.out.println("  H -->" + H + "   S --
>" + S + "   V -->"     + V);

                        if (((H >= 2.0 && H <= 6.0) || (H >= 350.0 &&
H <= 360.0)
                                    && S >= 0.83 && S <= 0.85 && V >=
0.45 && V <= 1.0)
                                    || ((R >= 150) && (R <= 255) && (G
>= 0) && (G <= 75)
                                                && (B >= 0) && (B <
75))) {
                            //System.out.println("RED");
                            ripe++;
                        }

                        if (((H >= 20.0 && H <= 50.0) && (S >= 0.65 &&
S <= 1.00) && (V >= 0.65 && V <= 1.00))
                                    || ((R >= 233 && H <= 255) && (R
>= 69 && R <= 160) && ((G >= 0 && G <= 122)))) {
                            //System.out.println("ORANGE");
                            medripe++;
                        }
                        //
                        if (R == 178 && G == 236 && B == 93)
                            unripe++;
                        if (R == 124 && G == 252 && B == 0)
                            unripe++;
                        if (R == 102 && G == 255 && B == 0)
                            unripe++;
                        if (R == 172 && G == 225 && B == 175)
                            unripe++;
                        if (R == 178 && G == 190 && B == 181)
                            unripe++;
                        if (R == 163 && G == 193 && B == 173)
                            unripe++;
                        if (R == 147 && G == 197 && B == 114)
                            unripe++;
                        if (R == 133 && G == 187 && B == 101)
                            unripe++;
                        if (R == 3 && G == 192 && B == 60)
                            unripe++;
                        if (R == 120 && G == 134 && B == 107)
                            unripe++;
                        if (R == 115 && G == 134 && B == 120)
                            unripe++;
                        if (R == 0 && G == 128 && B == 0)
                            unripe++;
```

```
                        if (R == 19 && G == 136 && B == 8)
                              unripe++;

                        if ((R >= 175) && (R <= 250) && (G >= 100) &&
(G <= 255)
                                    && (B >= 0) && (B < 10)) {
                              unripe++;

                              //System.out.println(" Unripe ");
                        }

                        // if (!((H == 60.0) && (S == 0.0) && (V ==
1.0)) || ((H == 0.0)
                        // && (S == 0.0)&& ((V >= 0.30) && (V <=
0.40)))) {
                        // total++;
                        //
                        //
                        // }
                        if (!((R == 255 && G == 255 && B == 255))
                                    || (((H == 60.0) && (S == 0.0) &&
(V == 1.0)) || ((H == 0.0)
                                          && (S == 0.0) && ((V
>= 0.30) && (V <= 0.40))))) {
                              total++;
                        }

                  }
            }

            int totalpercent = unripe + medripe + ripe;

            String a = "UNRIPE: " + ((unripe * 100) / totalpercent) +
"%  MEDRIPE: "
                        + ((medripe * 100) / totalpercent) + "%  RIPE:
"
                        + ((ripe * 100) / totalpercent) + " %    ";
                        //+ "    Threshold :" +
otsuTreshold(newphoto4);
            // TODO Auto-generated method stub
            return a;
      }

      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.agribotrealtime);

            button = (Button) findViewById(R.id.button1);
            imageView = (ImageView) findViewById(R.id.imageview);
            textView = (TextView) findViewById(R.id.state);
            // imageView.getLayoutParams().height = 50;

            button.setOnClickListener(new OnClickListener() {

                  public void onClick(View v) {
                        // TODO Auto-generated method stub
                        Intent i = new Intent(

      android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
                        startActivityForResult(i, requestcode);
```

```
                }
            });

        }

        @Override
        protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
            if (requestCode == requestcode) {
                photo = (Bitmap) data.getExtras().get("data");

                newphoto = doGreyscale(photo);
                newphoto2 = binarize(newphoto, photo);
                String a = "";
                a = findcolor(newphoto2);

                imageView.setLayoutParams(new
LinearLayout.LayoutParams(
                        LinearLayout.LayoutParams.WRAP_CONTENT,

        LinearLayout.LayoutParams.WRAP_CONTENT));

                imageView.getLayoutParams().height = 300;
                imageView.getLayoutParams().width = 300;
                imageView.setImageBitmap(newphoto2);
                textView.setText(a);


                  //Sending sms
                PendingIntent pi =
PendingIntent.getActivity(this,0,data, 0);
                SmsManager sms = SmsManager.getDefault();
                  sms.sendTextMessage("8097511597", null, a,
pi,null);

            }


        }
}
```

5.3 DEPLOYMENT

1. Overview

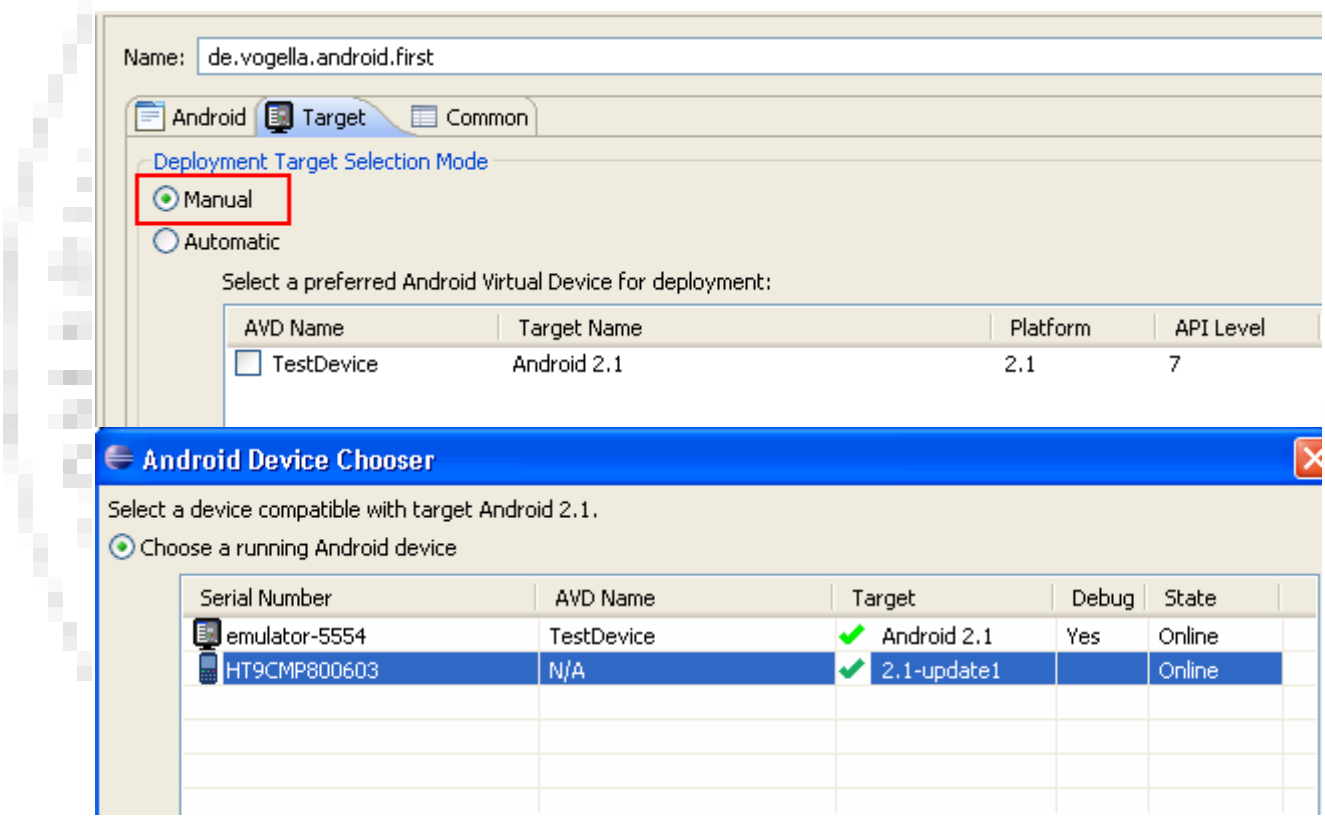
In general there are you restrictions how to deploy an Android application to your device. You can use USB, email yourself the application or use one of the many Android markets to install the application. The following describes the most common ones.

2. Deployment via the ADT

Turn on "USB Debugging" on your device in the settings. Select in the settings Applications > Development, then enable USB debugging. You also need to install the driver for your mobile phone. Please note that the Android version you are developing for must be the installed version on your phone.

To select your phone, select the "Run Configurations", select "Manual" selection and select your device.

Fig. 15: Deployment via ADT



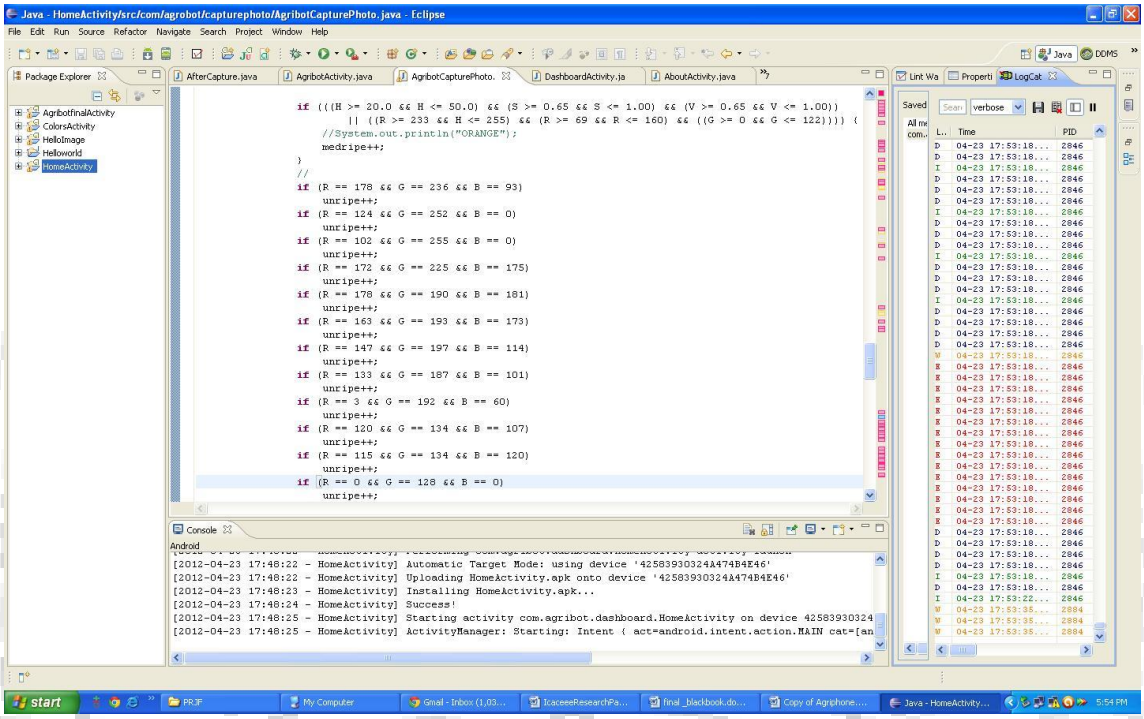
3. Via external sources

Android allow to install applications also directly. Just click on a link which points to an .apk file, e.g. in an email attachment or on a webpage. Android will prompt you if you want to install this application.

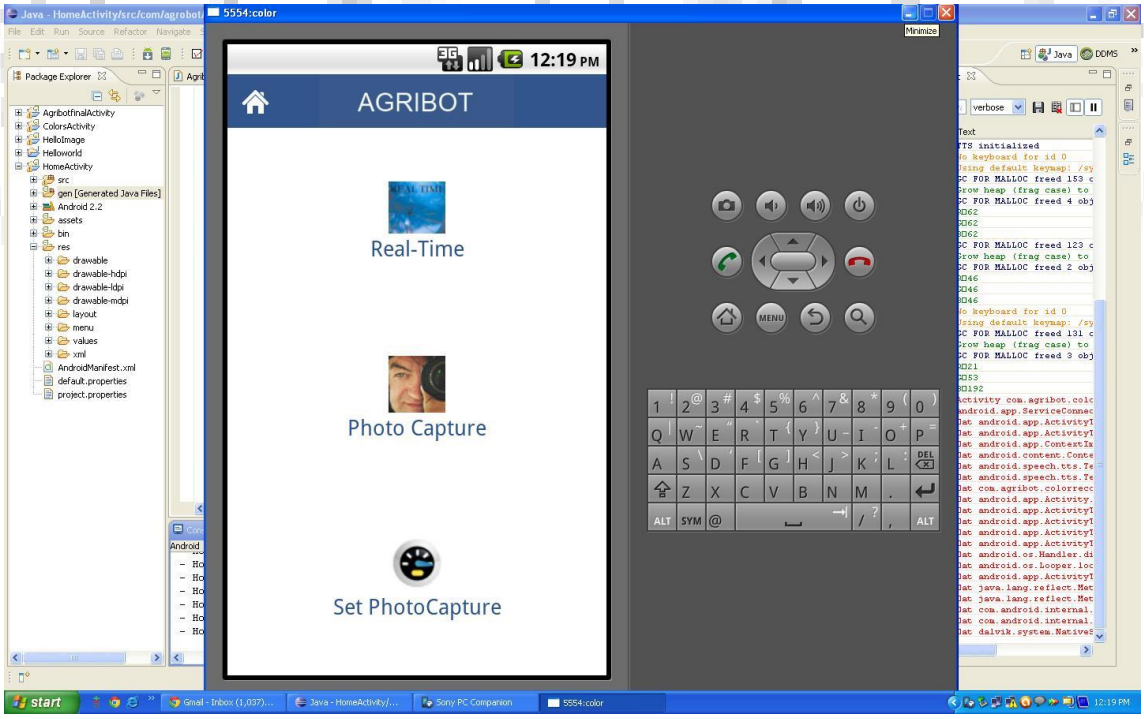
This requires a setting on the Android device which allows the installation of non-market application. Typically this setting can be found under the "Security" settings.

5.4 SNAPSHOTS:

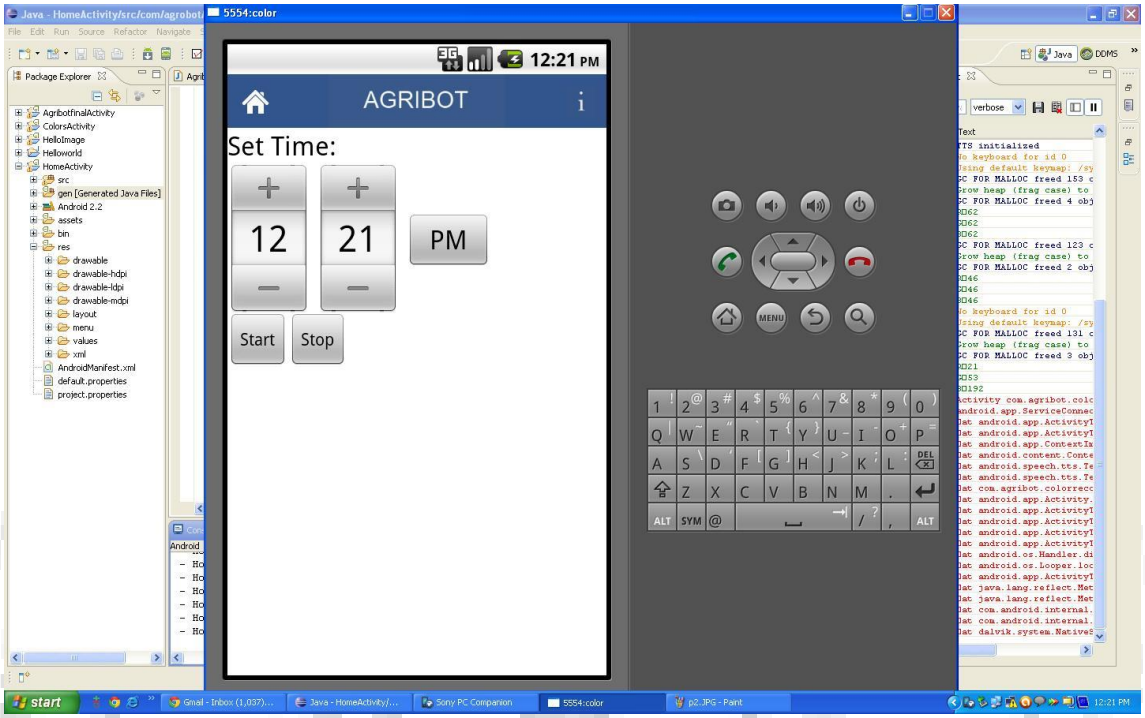
Snapshot 1: Eclipse Environment



Snapshot 2: Home Screen of project



Snapshot 3: Set Alarm for capturing photo



Snapshot 4: Capturing a photo

Snapshot 5: Real Time Application


Sony Ericsson
XPERIA
01/20/2006

CHAPTER 6.
TESTING PROCEDURE

6.1 OBJECTIVES OF TESTING

- Testing is the process of executing the program with the intent of finding an error.
- A good test is a one that has high probability of finding an undiscovered error.
- A successful test is the one that uncovers a yet undiscovered error.

6.2 TESTING TECHNOLOGY USED

The two types of error which are possible in program are syntactical errors and logical error.

- Syntactical errors: these errors are easily debugged.
- Logical errors: These errors are difficult to debug they do not cause the program to stop running but give incorrect outputs , which can only be understood and debugged through testing.

This project has been tested on the following two testing technologies.

- **Unit testing:**

Unit testing deals with testing a single module at a time. Care is taken while designing modules to keep them independent of each other. This would test the interaction of many functions but confine the test within one unit. Under unit testing program it can be broken into several functions, independent units can be designed separately from each other.

Test plan and Test Results:

Test Case No:	**Test Case**	**Result Expected**	**Actual Result**
1	Alarm Service	It must be able to set time	Time is set
2	Capture photo	Photo must be captured at specified time	Photo is captured At specified time
3	Saving photo to sd card	Photo must be saved automatically to sd card	Photo is saved to sd card

4	Processing of photo	Photo must be processed in the phone itself	Photo is processed in the phone
5	Extracting RGB values	Extract RGB value of each pixel	RGB value of each pixel is extracted
6	Converting RGB values to HSV	RGB values must be converted to HSV model	RGB values converted to HSV model
7	Calculating Ripe, Unripe, Medium ripe and Total no of pixels	Ripe, Unripe, Medium ripe and Total no of pixels must be calculated	Ripe, Unripe, Medium ripe and Total no of pixels are calculated
8	Finding percentage of ripe, medium ripe and ripe part	Find percentage of ripe, medium ripe and ripe part	Percentage of ripe, medium ripe and ripe part is calculated
9	Sending SMS	Send SMS to the user	SMS is sent

Table 2: Test Plan and Test Results I

- White-box testing:-

It is also known as code testing. Under white box testing, test is carried out in order to run every instruction present in the program. White box testing is used to test every path in the program

- Black-box testing:

It is also known as functional testing. Under this testing the program is considered as a black box and is tested to see whether it performs as per defined specifications.

- **Integration testing:**

After unit testing is made clear that the functionality separate units are working as per specifications. In this testing units are tested together to check errors in the code.

Test plan and Test Results:

Test Case No.	**Test Case**	**Result Expected**	**SMS Recieved**
1	RIPED TOMATO	UNRIPE:0% MEDRIPE:0% RIPE:100%	UNRIPE: 0% MEDRIPE:0% RIPE:100%
2	UNRIPE TOMATO	UNRIPE:80% MEDRIPE:20% RIPE:0%	UNRIPE:70 % MEDRIPE:30% RIPE:0%
3	MEDIUM RIPE TOMATO	UNRIPE:0% MEDRIPE:100% RIPE:0%	UNRIPE:0 % MEDRIPE:99% RIPE:0%
4	RIPED AND UNRIPED TOMATO	UNRIPE:20% MEDRIPE:10% RIPE:70%	UNRIPE: 20% MEDRIPE:10% RIPE:70%
5	RIPED AND MEDIUM RIPED TOMATO	UNRIPE:0% MEDRIPE:40% RIPE:70%	UNRIPE: 0% MEDRIPE:33% RIPE:67%
6	MEDIUM RIPED AND UNRIPED TOMATO	UNRIPE:35% MEDRIPE:50% RIPE:15%	UNRIPE: 33% MEDRIPE:54% RIPE:13%
7	RIPED, MEDIUM RIPED AND UNRIPED TOMATO	UNRIPE:20% MEDRIPE:60% RIPE:20%	UNRIPE: 13% MEDRIPE:67% RIPE:18%

Table 3: Test Plan and Test Results II

CHAPTER 7.
CONCLUSION

Thus we have successfully done with our project. We have implemented the application software for android based phone which can detect ripeness degree of either single tomato fruit or combined ripeness degree of multiple tomatoes.

We have also implemented automation in the project with the help of alarm clock service of android phone.

Thus, this application is able to automatically capture photos of tomatoes and save it in sd card for processing it.

We have also implemented SMS service which delivers SMS regarding the ripeness degree of fruit in terms of percentage.

We have also implemented real time fruit ripeness detection module which can give status of ripeness for few pixels of fruit which are in the range of camera.

The project we have build will be useful for greenhouse owners after some modifications and enhancement in coding.

CHAPTER 8.
FUTURE WORK

The next step of application development on mobile platform for greenhouse is to implement and test more complex image processing applications that contain photos captured from greenhouse for variety of plants.

Also it can be further improved to detect the various diseases of greenhouse plants.
Apart from this, agriphone can be used in robotic fruit harvesting.
Also in other applications of greenhouse which requires vision as a sensor.

There is always chance to improve any system as research & development is an endless process. Our system is no exception to this phenomenon.
The following improvements can be done:

- Effect of intensity of light can be reduced using advanced algorithms and image processing techniques.
- This application can be developed for any other type of fruit.

CHAPTER 9.
BIBLIOGRAPHY

- [1] Izzatdin Abdul Aziz, Mohd Hilmi Hasan, Mohd Jimmy Smail, Mazlina Mehat, Nazleeni Samiha Haronr , "Remote Monitoring in Agricultural Greenhouse Using Wireless Sensor and Short Message Service (SMS)", International Journal of Engineering & Technology IJET Vol: 9 No: 9
- [2] Guo Feng; Cao Qixin and Nagata Masateru
  Research Institute of Robotics, Shanghai Jiao Tong University, Shanghai China Blob based Singular Strawberry Detection Algorithm
- [3] Hayashi S; Sakaue O (1996). Tomato harvesting by robotic system. ASAE Paper No. 96-3067
- [4] Yousef Al Ohali , "Computer vision based date fruit grading system: Design and implementation" Journal of King Saud University –Computer and Information Sciences
- [5] Ruiz, L.A., Molto, E., Juste, F., Pla, F., Valiente, R., "Location and characterization of the stem calyx area on oranges by computer vision", *Journal of Agricultural Engineering Research*, 64, pp. 165-172, 1996.
- [6] Kondo, N., Nishitsuji Y, Ling, P. & Ting, K.C. 1996. Visual Feedback Guided Robotic Cherry Tomato Harvesting. Transactions of the ASAE, 39 (6), 2331-2338.
- [7] "Fruit Detection using Improved Multiple Features based Algorithm. International Journal of Computer Applications (0975 – 8887)" Volume 13– No.2, January 2011
- [8] Hosna Mohamadi Monavar, Reza Alimardani, Mahmoud Omid "Detection of red ripe tomatoes on stem using Image Processing Techniques" Journal of American Science, 2011;7(7)
- http://www.vogella.com/articles/Android/article.html#overview_adt
- http://www.stanford.edu/class/ee368/Android/Tutorial-1-Basic-Android-Setup-Windows.pdf
- http://developer.android.com/guide/developing/device.html