Governors State University
OPUS Open Portal to University Scholarship

All Capstone Projects

Student Capstone Projects

Fall 2015

# Future SMS

Varun Chowdhary Enjum
*Governors State University*

Suresh Kumar Kurapati
*Governors State University*

Umakanth Vellanki
*Governors State University*

# Table of Contents

# 1. Project Description

## 1.1 Project Abstract

The main purpose of this project is automatic SMS messages sending. It provides flexible scheduling system to the user. That means one can schedule a message according to the required date and tie.

## 1.2 Competitive Information

One may forget to send a SMS regarding birthday wishes or anniversary wishes or any other important information. To overcome such problems an android application called Future SMS has been developed wherein a SMS will be automatically sent on the selected date and time to desired recipient.

One can schedule a SMS by simply opening Future SMS application. Click on the button 'Schedule' where every entry specifies recipient's number, date and time of your transmission and the text message to be sent. Then just click on the button 'Done' to schedule your SMS. To add a new entry, follow the same procedure. One can schedule any number of messages as per requirements.

## 1.3 Relationship to Other Applications/Projects

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design.

**SYSTEM SPECIFICATIONS**

**Processes Involved**

- Set up user interface process.
- SMS process.
- Remainder process.
- Archiving and Backup process.

## 1.4 SYSTEM COMPONENTS

Modules Involved:-

There are five modules in the project. The modules which are being dealt in the project are given below:

**1.** UI Module:

In this module we can design the screen according to your application requirements.

**2.** SMS Module:

In this module we can implement the business logic for how to send SMS based on Date and Time.

**3.** Remainder Module:

In this module we can schedule the date and time for future purpose with message like wishes, any important remainders or updates etc.

**4.** Archiving and backup Module:

Allows Admin users to Archive/Backup old data on the system.

## 1.5 Assumptions and Dependencies

1) Describe any assumptions made
2) Features a highly customizable message scheduling screen
3) Acknowledge a birthday, anniversary, or recurring special event
4) In future SMS system all of the implementations were confined to a single J2SE file. It was relatively straightforward to make sure that classes from these J2SE files were available at runtime.

## 1.6 Future Enhancements

The current feature in FUTURE SMS APPLICATION may need to further change and improvement to support the present technology. Therefore, there are some recommendations that might be done in future, such as:

Add more features to continue the system enhancement. There are several improvements that can be made such as sending stickers along with the text message, sending a multimedia message.

To provide ease for the user by developing an inbuilt keypad that reduces the effort to type whole word and the chance of spelling mistakes.

## 1.7 Definitions and Acronyms

- Future SMS application is a simple program which allows scheduling and storing a message using SMS scheduler. It provides comfort to the user as he can schedule a message any time and not think of it later when it is actually to be sent.
- Our goal is to be compatible with as many devices as possible
- Future SMS, you don't need to duck out of work, class, or personal time

# *2.* Technical Description

## 2.1 Project/Application Architecture

The first activity in the phase is studying the existing system and other is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of existing running system is also difficult, improper understanding of present system can lead diversion from solution.

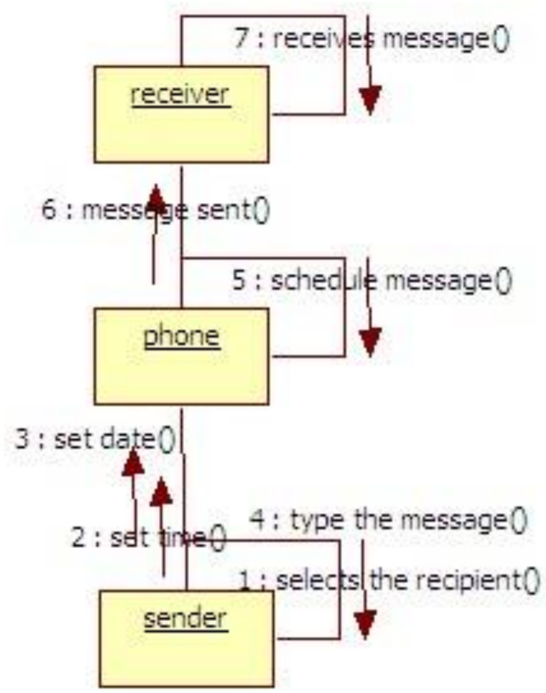



**Fig 1: Collaboration Diagram**

## 2.2 Project/Application Information flows

In the existing system one could send a normal text message only at that particular instant when he wants to. The user had no choice to save the message and schedule it to a required date and time he loved to send as SMS.

**Disadvantages**

The following are the disadvantages faced in existing system:

- No particular information and updates could be sent at the right time.
- There is no option to schedule messages for coming events in future.
- There is no implementation of business logic for how to send SMS based on date and time.

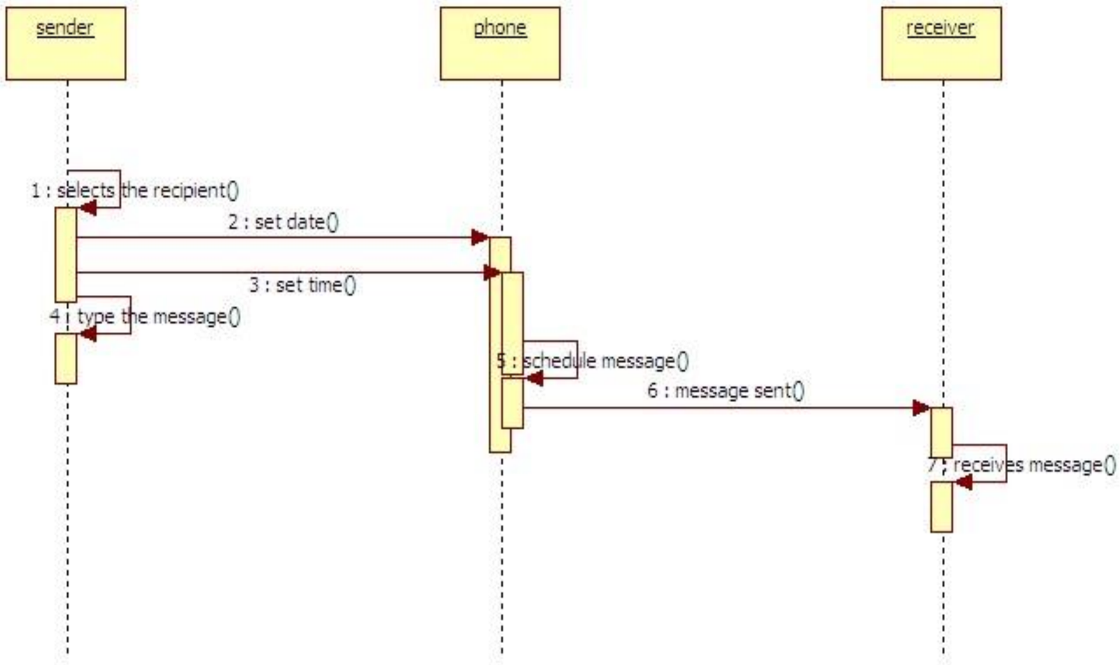



**Fig 2: Sequence Diagram**

## 2.3 Capabilities

The goal of this project is to develop SMS application that will utilize the SMS technology and add additional features to it like one can schedule a message to a particular date and time according to his requirement. One may forget to wish his friend's birthday or unable to send marriage wishes or any particular information and updates and fails to send message at the right time.

**PROPOSED SYSTEM**

Keeping in view the above problem an Android application called Future SMS has been developed. By using this application one can send a message on the right time by simply scheduling it using SMS scheduler.

### Advantages

The following objectives were highlighted while designing the above application:

- Sending information and updates at the right time.
- Implementation of business logic for how to send SMS based on date and time.

- Scheduling any number of messages according to the required date and time.

## 2.4 Risk Assessment and Management

Testing is one of the most important phases in the software development activity. In software development life cycle (SDLC), the main aim of testing process is the quality; the developed software is tested against attaining the required functionality and performance.

During the testing process the software is worked with some particular test cases and the output of the test cases are analyzed whether the software is working according to the expectations or not.

The success of the testing process in determining the errors is mostly depends upon the test case criteria, for testing any software we need to have a description of the expected behavior of the system and method of determining whether the observed behavior confirmed to the expected behavior.

# 3. Project Requirements

## 3.1 Identification of Requirements

Requirements Specification plays an important role in creating quality software solutions; Requirements are refined and analyzed to assess the clarity. Specification is basically a representation process.

Requirements are represented in a manner that ultimately leads to successful software implementation. Each requirement must be consistent with the overall objective. The development of this project deals with the following requirements:

- Hardware requirement
- Software requirements
-

### HARDWARE REQUIREMENTS

The selection of hardware is very important in the existence and proper working of any software. In the selection of hardware, the size and the capacity requirements are also important.

**Table 1 Hardware Requirements**

| CONTENT | DESCRIPTION |
|---|---|
| HDD | 40 GB Recommended |
| RAM | 1 GB Recommended |
| DEVICE | Android Phone |

## SOFTWARE REQUIREMENTS

One of the most difficult tasks is that, the selection of the software, once system requirement is known by determining whether a particular software package fits the requirements.

**Table 2 Software Requirements**

| CONTENT | DESCRIPTION |
|---|---|
| Operating System | Android, Linux, Windows XP |
| Technologies | J2SE, ADT plug-in |
| IDE | Android SDK,Android Emulator |

## 3.2 Operations, Administration, Maintenance and Provisioning (OAM&P)

Initially the language was called as "oak" but it was renamed as "Java" in 1995. The primary motivation of this language was the need for a platform-independent (i.e., architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

**Java is a programmer's language.**

1. Java is cohesive and consistent.

2. Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.

3. Finally, Java is to Internet programming where C was to system programming.

**Features of Java Security**

Every time you download a "normal" program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scan them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against. This type of program can gather private information, such as credit card numbers, bank account balances, and passwords. Java answers both these concerns by providing a "firewall" between a network application and your computer. When you use a Java-compatible Web browser, you can safely download Java applets without fear of virus infection or malicious intent.

**Portability**

For programs to be dynamically downloaded to all the various types of platforms connected to the Internet, some means of generating portable executable code is needed .As you will see, the same mechanism that helps ensure security also helps create portability. Indeed, Java's solution to these two problems is both elegant and efficient.

## The Byte Code

The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for byte code. Translating a Java program into byte code helps makes it much easier to run a program in a wide variety of environments. The reason is, once the run-time package exists for a given system, any Java program can run on it.

Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of byte code into native code. Sun has just completed its Just In Time (JIT) compiler for byte code. When the JIT compiler is a part of JVM, it compiles byte code into executable code in real time, on a piece-by-piece, demand basis. It is not possible to compile an entire Java program into executable code all at once, because Java performs various run-time checks that can be done only at run time. The JIT compiles code, as it is needed, during execution.

## Java Virtual Machine (JVM)

Beyond the language, there is the Java virtual machine. The Java virtual machine is an important element of the Java technology. The virtual machine can be embedded within a web browser or an operating system. Once a piece of Java code is loaded onto a machine, it is verified. As part of the loading process, a class loader is invoked and does byte code verification makes sure that the code that's has been generated by the compiler will not corrupt the machine that it's loaded on. Byte code verification takes place at the end of the compilation process to make sure that is all accurate and correct. So byte code verification is integral to the compiling and executing of Java code.

Java programming uses to produce byte codes and executes them. The first box indicates that the Java source code is located in a. Java file that is processed with a Java compiler called javac. The Java compiler produces a file called a. class file, which contains the byte code. The Class file is then loaded across the network or loaded locally on your machine into the execution environment is the Java virtual machine, which interprets and executes the byte code.

## Java Architecture

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

## Compilation of code

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine.

Compiling and Interpreting Java Source Code

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be Intel Pentium Windows 95 or SunSARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through Internet and run the Applets.

## Simple

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the object oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

## Object-Oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

### Robust

The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time.

Java virtually eliminates the problems of memory management and de-allocation, which is completely automatic. In a well-written Java program, all run time errors can –and should –be managed by your program.

## 3.2 Security and Fraud Prevention

Replace this section with description and requirement to address possible internal and external security issues.

### Release and Transition Plan

It has been a great pleasure for us to work on this exciting and challenging project. This project helped us provide a practical knowledge on programming in JAVA, J2SE and ADT plug-in. This will provide better opportunities and guidance in future developing projects independently. It also provides knowledge about latest technology used in developing Android based application that will be great demand in future. This will provide better opportunities and guidance in future developing projects independently.

## 4. Project Design Description

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word "Quality". Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer's view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

## UML DIAGRAMS

The Unified Modeling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

**User Model View**:

This view represents the system from the user's perspective. The analysis representation describes a usage scenario from the end-users perspective.

**Structural model view**:

In this model the data and functionality are arrived from inside the system. This model view models the static structures.

**Behavior Model View:**

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

**Implementation Model View:**

In this the structural and behavioral as parts of the system are represented as they are to be built.

**Environmental Model View:**

In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

Use case Diagrams represent the functionality of the system from a user's point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.

SYSTEM DESIGN

Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer …etc., or another system like central database.

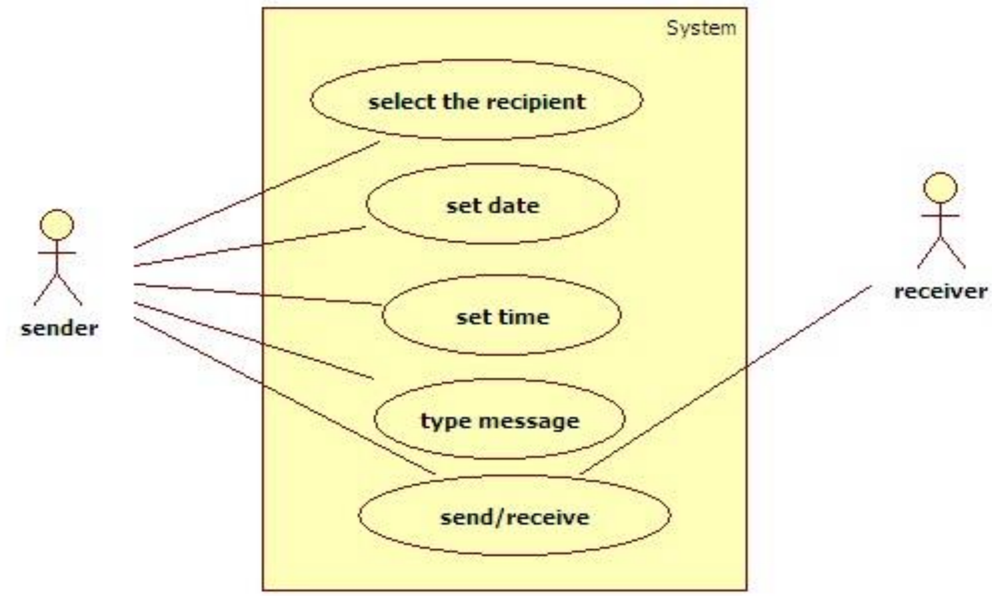



**Fig 3: Use case Diagram**

## 4.1 Project Internal/external Interface Impacts and Specification

Lot of advances can be seen these days in the field of smart phones. As the number of users is increasing day by day, facilities are also increasing. Starting with simple phones which were made just to make and receive calls. Now we have phones which can even access GPS, GPRS, Wi-Fi, NFC and lot of other cool and advanced features which you cannot even imagine.

So in this Mobile world of such complications Android is one of those operating system platforms which made it easy for manufacturers to design top class phones.

### What is Android?

You might have seen Windows, Linux and Mac operating systems which are made for computers. Windows is the most popular operating system on computers. So if you know about it then it is easy for you to get an answer for what is android.

Android is also an operating system developed by Google. Basically it was started by some other company which was taken by Google. Google improved the operating system and made it a open source platform. It was widely adapted over the world. As it is open source it is so popular amongst the smart phones. Android OS can also be used on tablet PCs.

Android is based on **LINUX** and offers you a great deal of customization in widgets and over millions of apps. Most of them are free of cost and can be installed on your phone just by clicking on install tab of the respective app in the Google Play Store which comes along with the Android Phone.

One of the most widely used mobile-OS these days is **ANDROID. It** is a software bunch comprising not only operating system but also middle ware and key applications. Android Inc was founded in Pa lo Alto of California, U.S. by Andy Rubin, Rich miner, Nick sears and Chris White in 2003. Later Android Inc. was acquired by Google in 2005. After original release there have been number of updates in the original version of Android.

### Which phone Manufacturer use Android?

Android is an open source platform which can be used by any phone manufacturers of the world unlike other operating systems for mobile phones like i OS (Operating System by apple for i Phone, i Pad and other i Devices.). Symbian is owned by Nokia and it comes only on Nokia Handsets. Android can be used by any manufacturer. So that if the latest research is to be believed over half of the smart phones in USA run on android.

Android is one the hottest mobile operating systems available today. Samsung is the largest manufacturer of android phones and tablets. LG, HTC, Sony, are other top manufacturers of android phones and tablets. Some local manufacturers like Micro-max, Karbon, Hawaii, also use android Phones on their portable devices.

## 5. MY ECLIPSE

### Setting up Development Environment in Eclipse

It is very easy to set up Android Development Environment on Eclipse. First of all you need to download the tools and software. You must download followings.

Java SE Development Kite (JDK 5 or newer)

Eclipse IDE for Java Developer

Android Software Development Kit

### Setup Android Software Development Kit (SDK):

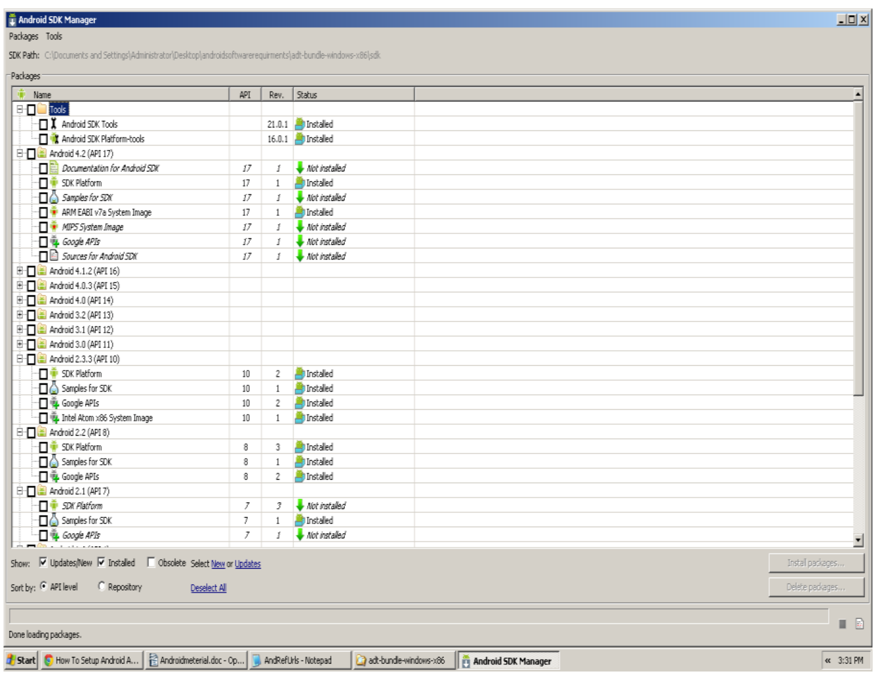



**Fig 4: Selecting android SDK platform**

- Select Available packages from the left pane and then select any android sdk platform you wish to develop from the right pane, it will take time to complete.
- Click Install Selected.

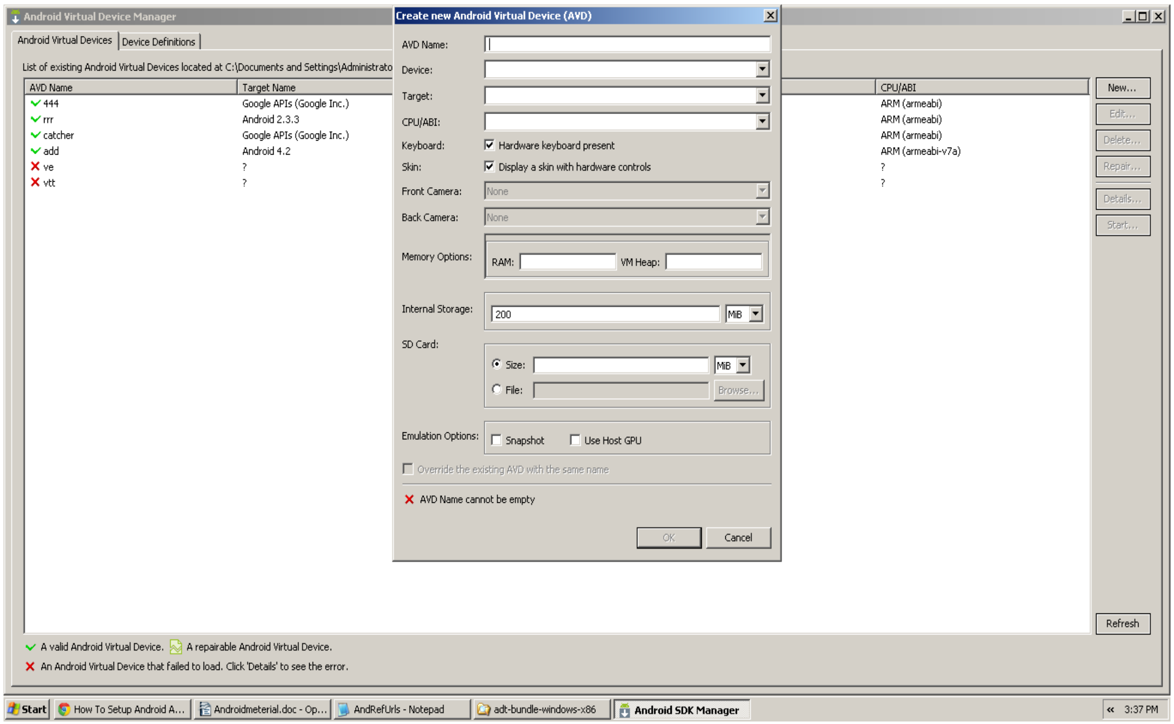



**Fig 5: Create new Android Virtual Device**

- Enter a name to your Android Virtual Device
- Select target Android version from the Target drop down box
- Give the size of SD card and then click Create AVD
- It will take time to create Android Virtual Device (AVD)

**Configure Installed Android with Eclipse:**

Run the Eclipse......

- First run of Eclipse it will ask the default workplace, you must be mention the default workplace folder
- Go to the Help -- Install New Software
- Click Add button in install window to install Android Developer Tool
- Name it as Android
- Add Location as http://dl-ssl.google.com/android/eclipse/ and do OK
- After click OK, in the next popup box Select Developer Tools and click Next and accept license agreement and finish it

- Eclipse IDE will gives you a warning message saying Your installation software that contains unsigned content ...... just OK and restart your Eclipse IDE
- Now you need to give the Android SDK location to the popup which will be display when Eclipse IDE restarts
- When the Eclipse IDE starts there will be a configuration popup which will ask the Android SDK location
- Select Use existing SDK and then browse the Android SDK location and finish.
- Now your Android development Environment is ready for development. Start your first Android Application in Eclipse Hello world Android example.

Steps:

**Create Android Project**

In Eclipse, select "File -> New -> Project….", "Android Application Project", and input your application detail like Name, Target etc. Eclipse will create all the necessary Android project files and configuration then click Next--------Next---------Next--------------Next------------- --Finish

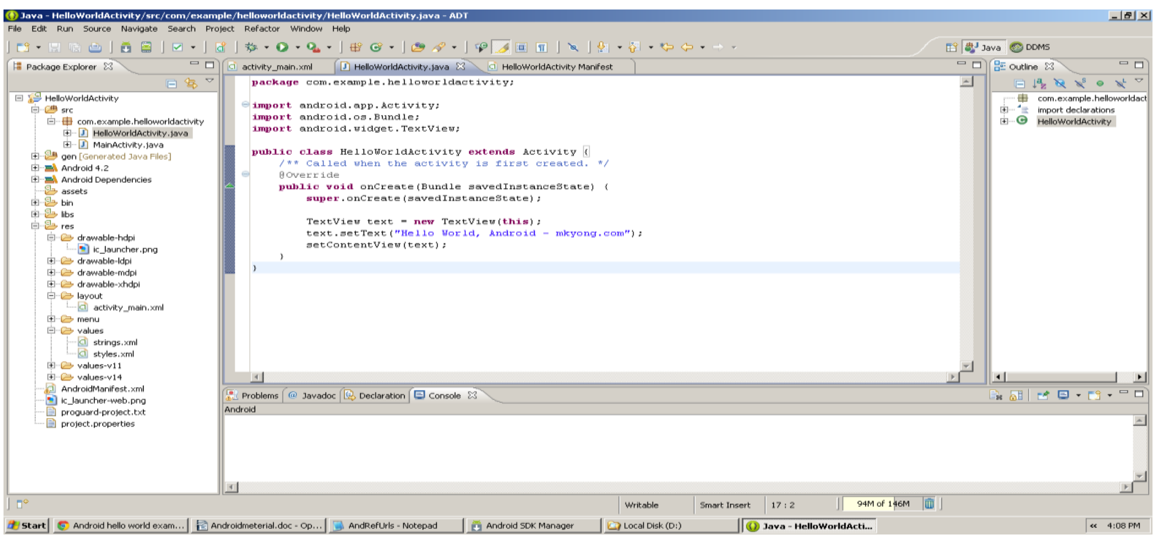



**Fig 6:` Example for Android application project**

**Hello World**

**Output**

Select Project---------In Menu Project-----------clean--------ok----------Right Click on Project----------Run-------Run an Android Application.

## 6. Project Design Units Impacts

### 6.1 Functional Area/Design Unit A

Implementation literally means to put into effect or carry out. The system implementation phase if the software deals with the translation of the design specifications into the source code. The ultimate goal of the implementation is to write the source code and the internal documentation so that it can be verified easily. The code and documentation should be written in a manner that eases debugging, testing and modification. System flow charts, sample run on packages, sample output etc., is part of implementation.

- Clarity and simplicity of the code.
- Minimization of hard coding.

## SAMPLE CODE

### MAIN ACTIVITY

```
package com.example.futuresms;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.HashMap;
import java.util.Map;
import android.app.Activity;
import android.app.AlarmManager;
import android.app.AlertDialog.Builder;
import android.app.PendingIntent;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.AdapterView.OnItemLongClickListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
public class MainActivity extends Activity {
public static final int MAIN_ACTIVITY_CONSTANT = 1;
        DateFormat sdf = SimpleDateFormat.getDateTimeInstance();
        DateFormat dateFormat = SimpleDateFormat.getDateInstance();
        DateFormat timeFormat =  SimpleDateFormat.getTimeInstance();
        Button schedule;
        ListView list;
private final Map<String, String> displayDataMap = new HashMap<String, String>();
                private  ArrayAdapter<String> adapter;
```

```
			@Override
			protected void onCreate(Bundle savedInstanceState) {
			super.onCreate(savedInstanceState);
			setContentView(R.layout.activity_main);
			TextView welcome = (TextView)findViewById(R.id.welcome_one);
			try{
			list = (ListView)findViewById(R.id.list);}
			catch(Exception e)
			{
			welcome.setText("beep beep. error");
			}
		schedule = (Button)findViewById(R.id.schedule);
			schedule.setOnClickListener(new OnClickListener(){
		@Override
			public void onClick(View v) {
			Intent fireScheduleActivity = new Intent(MainActivity.this, Scheduler.class);
			startActivityForResult(fireScheduleActivity, MAIN_ACTIVITY_CONSTANT);
			// TODO Auto-generated method stub
			}
			});
	adapter = new ArrayAdapter<String>(this, android.R.layout.simple_selectable_list_item);
		list.setAdapter(adapter);
		list.setOnItemClickListener(new OnItemClickListener(){
		@Override
		public void onItemClick(AdapterView<?> arg0, View view, int position,
		long id) {
		String key = adapter.getItem(position);
		String data = displayDataMap.get(key);
		long time = Long.valueOf(data.split("aqlpzaml")[0]);					String
	addressee = data.split("aqlpzaml")[1];
		String text = data.split("aqlpzaml")[2];
		String formattedTime = timeFormat.format(time);
		String formattedDate = dateFormat.format(time);
	String dialogData = "To: " +  addressee + "\n" + "Date: " + formattedDate + "\n" + "Time: " + formattedTime +
	"\n\n" + "SMS: " + "\n" + text;
	ShowMessage fragment = ShowMessage.getDialogFragment(dialogData);
		fragment.show(getFragmentManager(), "dialog");
		}
		});
		list.setOnItemLongClickListener(new OnItemLongClickListener(){
		@Override
		public boolean onItemLongClick(AdapterView<?> arg0, View view,
		int position, long id) {
		final String key = adapter.getItem(position);
		String data = displayDataMap.get(key);
		final String addressee = data.split("aqlpzaml")[1];
		long time = Long.valueOf(data.split("aqlpzaml")[0]);
		final int finalID = Integer.valueOf(data.split("aqlpzaml")[3]);
		final String dateTime = sdf.format(time);
		Builder dialog = new Builder(view.getContext());
		dialog.setTitle("! Delete Message" );
		dialog.setMessage("Delete this message ? ");
	dialog.setPositiveButton("Delete", new DialogInterface.OnClickListener() {
		@Override
		public void onClick(DialogInterface dialog, int which) {
```

```
        // TODO Auto-generated method stub
        displayDataMap.remove(key);
        adapter.remove(addressee + " " + dateTime);
        adapter.notifyDataSetChanged();
        Intent cancelIntent = new Intent(getApplicationContext(),
        SendSMS.class);
PendingIntent pdi = PendingIntent.getActivity(getApplicationContext(), finalID, cancelIntent,
        PendingIntent.FLAG_ONE_SHOT);
AlarmManager cancelManager = (AlarmManager)getSystemService(Context.ALARM_SERVICE);
        cancelManager.cancel(pdi);
Toast toast =Toast.makeText(getApplicationContext(), "SMS cancelled", Toast.LENGTH_LONG);
        toast.show();
        }
        });
dialog.setNegativeButton("Back", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
        // TODO Auto-generated method stub
        dialog.cancel();
        }
        });
        dialog.create().show();
        // TODO Auto-generated method stub
        return false;
        }
        });
        }
        @Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        // TODO Auto-generated method stub
        super.onActivityResult(requestCode, resultCode, data);
        if(resultCode == Scheduler.SCHEDULER_CONSTANT)
        {
        String addressee = data.getStringExtra("smsAddressee");
        String smsText = data.getStringExtra("smsData");
        long time = data.getLongExtra("dateTime", 0);
        int uniqueID = data.getIntExtra("requestCode", 0);
        String dateTime = sdf.format(time);
String finalData = addressee + " " + dateTime;
        adapter.add(finalData);
displayDataMap.put(finalData, time + "aqlpzaml" + addressee + "aqlpzaml" + smsText + "aqlpzaml" + uniqueID);
        adapter.notifyDataSetChanged();
}
        }
@Override
        public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
        }
}
```

## SCHEDULER

```
package com.example.futuresms;
import java.text.SimpleDateFormat;
import java.util.Calendar;
```

```
import android.app.Activity;
import android.app.AlarmManager;
import android.app.DatePickerDialog;
import android.app.PendingIntent;
import android.app.TimePickerDialog;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.text.format.DateFormat;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.TimePicker;
import android.widget.Toast;
public class Scheduler extends Activity {
	private static final String workCompleted = "Your SMS has been scheduled";
	public static final int SCHEDULER_CONSTANT = 2;
	AlarmManager newManager;
	EditText addressee;
	TextView dateView;
	TextView timeView;
	EditText smsText;
	Button scheduleSMSButton;
	Button scheduleTime;
	Button scheduleDate;
	Calendar cal;
	Calendar targetCal;
	Intent backIntent;
	SimpleDateFormat sfgDate = new SimpleDateFormat("dd.MM.yyyy");
	SimpleDateFormat sfgTime = new SimpleDateFormat(" HH:mm:ss");
	@Override
	protected void onCreate(Bundle savedInstanceState) {
	super.onCreate(savedInstanceState);
	setContentView(R.layout.scheduler);
	backIntent = getIntent();
	addressee = (EditText)findViewById(R.id.addressee);
	dateView = (TextView)findViewById(R.id.date);
	timeView = (TextView)findViewById(R.id.time);
	smsText = (EditText)findViewById(R.id.smsText);
	scheduleDate = (Button)findViewById(R.id.scheduleDate);
	scheduleTime = (Button)findViewById(R.id.scheduleTime);
	scheduleSMSButton = (Button)findViewById(R.id.scheduleTextButton);
	targetCal = Calendar.getInstance();
	targetCal.set(Calendar.SECOND, 00);
	dateView.setText(checkDate(targetCal.get(Calendar.DAY_OF_MONTH)) + "/" +
checkDate((targetCal.get(Calendar.MONTH )+ 1)) + "/" + checkDate(targetCal.get(Calendar.YEAR)));
	timeView.setText(checkDate(targetCal.get(Calendar.HOUR_OF_DAY)) + ":" +
checkDate(targetCal.get(Calendar.MINUTE)) + ":" + checkDate(targetCal.get(Calendar.SECOND)));
	scheduleDate.setOnClickListener(new OnClickListener(){
	@Override
	public void onClick(View v) {
```

```
cal = Calendar.getInstance();
int day = cal.get(Calendar.DAY_OF_MONTH);
int month = cal.get(Calendar.MONTH);
int year = cal.get(Calendar.YEAR);
DatePickerDialog dpd = new DatePickerDialog(Scheduler.this, d, year, month,
day);
dpd.show();
// TODO Auto-generated method stub
}
});
scheduleTime.setOnClickListener(new OnClickListener(){
@Override
public void onClick(View v) {
cal = Calendar.getInstance();
int hours = cal.get(Calendar.HOUR_OF_DAY);
int minutes = cal.get(Calendar.MINUTE);
TimePickerDialog tpd = new TimePickerDialog(Scheduler.this, t, hours, minutes,
DateFormat.is24HourFormat(getApplicationContext()));
tpd.show();
// TODO Auto-generated method stub
}
});
scheduleSMSButton.setOnClickListener(new OnClickListener(){
@Override
public void onClick(View v) {
long difference = targetCal.getTimeInMillis();
String smsData = smsText.getText().toString();
String smsAddressee = addressee.getText().toString();
int requestCode = (int)System.currentTimeMillis();
new SchedulingInProgress(difference, smsData, smsAddressee, requestCode).schedule();
Toast toast = Toast.makeText(Scheduler.this, workCompleted, Toast.LENGTH_LONG);
toast.show();
backIntent.putExtra("smsData", smsData);
backIntent.putExtra("smsAddressee", smsAddressee);
backIntent.putExtra("dateTime", difference);
backIntent.putExtra("requestCode", requestCode);
setResult(SCHEDULER_CONSTANT, backIntent);
finish();
// TODO Auto-generated method stub
}
});
}
Private DatePickerDialog.OnDateSetListener d = new DatePickerDialog.OnDateSetListener() {
@Override
public void onDateSet(DatePicker view, int thisyear, int monthOfYear,
int dayOfMonth) {
targetCal.set(Calendar.DAY_OF_MONTH, dayOfMonth);
targetCal.set(Calendar.MONTH, monthOfYear);
targetCal.set(Calendar.YEAR, thisyear);
long time = targetCal.getTimeInMillis();
dateView.setText(sfgDate.format(time));
// TODO Auto-generated method stub
}
};
private TimePickerDialog.OnTimeSetListener t = new TimePickerDialog.OnTimeSetListener() {
```

```
        @Override
        public void onTimeSet(TimePicker view, int hourOfDay, int minuteOfDay) {
        // TODO Auto-generated method stub
        targetCal.set(Calendar.HOUR_OF_DAY, hourOfDay);
        targetCal.set(Calendar.MINUTE, minuteOfDay);
        long time = targetCal.getTimeInMillis();
        timeView.setText(sfgTime.format(time));
        }
        };
        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.scheduler, menu);
        return true;
        }
        public String checkDate(int i)
        {
        String date;
        if(i<10)
        date = "0" + i;
        else
        date = ""+i;
        return date;
        }
        private class SchedulingInProgress{
long timeToTrigger;
        String data;
        String addressee;
        int code;
        public SchedulingInProgress(long difference, String smsData,
        String smsAddressee, int requestCode) {
        timeToTrigger = difference;
        data = smsData;
        addressee = smsAddressee;
        code = requestCode;
        // TODO Auto-generated constructor stub
        }
        private void schedule(){
        Intent fireSendSMSClass = new Intent(getApplicationContext(), SendSMS.class);
        fireSendSMSClass.putExtra("smsData", data);
        fireSendSMSClass.putExtra("smsAddressee", addressee);
PendingIntent pdi = PendingIntent.getActivity(getApplicationContext(), code, fireSendSMSClass,
        PendingIntent.FLAG_ONE_SHOT);
        newManager = (AlarmManager) getSystemService(Context.ALARM_SERVICE);
        newManager.set(AlarmManager.RTC_WAKEUP, timeToTrigger, pdi);
        }
        }
}
```

## SEND SMS

```
package com.example.futuresms;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.SmsManager;
```

```
import android.view.Menu;
public class SendSMS extends Activity {
	@Override
	protected void onCreate(Bundle savedInstanceState) {
	super.onCreate(savedInstanceState);
	setContentView(R.layout.send_sms);
	Intent intent = getIntent();
	String number = intent.getStringExtra("smsAddressee");
	String text = intent.getStringExtra("smsData");
	SmsManager manager = SmsManager.getDefault();
	manager.sendTextMessage(number, null, text, null, null);
	finish();
	}
	@Override
	public boolean onCreateOptionsMenu(Menu menu) {
	// Inflate the menu; this adds items to the action bar if it is present.
	getMenuInflater().inflate(R.menu.send_sms, menu);
	return true;
	}}
```

## SHOW MESSAGE

```
package com.example.futuresms;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import android.app.AlertDialog;
import android.app.Dialog;
import android.app.DialogFragment;
import android.content.DialogInterface;
import android.os.Bundle;
public class ShowMessage extends DialogFragment {
	String to;		String date;
	int time; String text;
	DateFormat dateFormat = SimpleDateFormat.getDateInstance();
	DateFormat timeFormat =  SimpleDateFormat.getDateTimeInstance();
	static ShowMessage getDialogFragment(String data){
	ShowMessage fragment = new ShowMessage();
	Bundle args = new Bundle();
	args.putString("data", data);
	fragment.setArguments(args);
	return fragment;
	}
	@Override
	public Dialog onCreateDialog(Bundle savedInstanceState) {
	// TODO Auto-generated method stub
	AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
	String data = getArguments().getString("data");
	builder.setTitle("Scheduled SMS:");
	builder.setMessage(data);
	builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
	@Override
	public void onClick(DialogInterface dialog, int which) {
	dismiss();
	// TODO Auto-generated method stub
	}
	});
	return builder.create(); }
```

### 6.2 Functional Overview

Since the errors in the software can be injured at any stage. So, we have to carry out the testing process at different levels during the development. The basic levels of testing are Unit, Integration, System and Acceptance Testing.

The Unit Testing is carried out on coding. Here different modules are tested against the specifications produced during design for the modules. In case of integration testing different tested modules are combined into sub systems and tested in case of the system testing the full software is tested and in the next level of testing the system is tested with user requirement document prepared during SRS.

There are two basic approaches for testing. They are

**Functional Testing:**

In Functional Testing test cases are decided solely on the basis of requirements of the program or module and the internals of the program or modules are not considered for selection of test cases. This is also called Black Box Testing.

## 7. Structural Testing:

In Structural Testing test cases are generated on actual code of the program or module to be tested. This is called White Box Testing.

**TESTING PROCESS**

A number of activities must be performed for testing software. Testing starts with test plan. Test plan identifies all testing related activities that need to be performed along with the schedule and guide lines for testing. The plan also specifies the levels of testing that need to be done, by identifying the different testing units. For each unit specified in the plan first the test cases and reports are produced. These reports are analyzed.

**Test Plan:**

Test plan is a general document for entire project, which defines the scope, approach to be taken and the personal responsible for different activities of testing. The inputs for forming test plans are

Project plan

Requirements document

System design

**Test Case Specification:**

Although there is one test plan for entire project test cases have to be specified separately for each test case. Test case specification gives for each item to be tested. All test cases and outputs expected for those test cases.

**Test Case Execution And Analysis:**

The steps to be performed for executing the test cases are specified in separate document called test procedure specification. This document specify any specify requirements that exist for setting the test environment and describes the methods and formats for reporting the results of testing.

**Unit Testing:**

Unit testing mainly focused first in the smallest and low level modules, proceeding one at a time. Bottom-up testing was performed on each module. As developing a driver program, that tests modules by developed or used. But for the purpose of testing, modules themselves were used as stubs. After the lower level modules were tested, the modules that in the next higher level those make use of the lower modules were tested. Each module was tested against required functionally and test cases were developed to test the boundary values.

**7.1 Impacts**

Integration testing is a systematic technique for constructing the program structure, while at the same time conducting tests to uncover errors associated with interfacing. As the system consists of the number of modules the interfaces to be tested were between the edges of the two modules. The software tested under this was incremental bottom-up approach.

Bottom-up approach integration strategy was implemented with the following steps.

- Low level modules were combined into clusters that perform specific software sub functions.
- The clusters were then tested.

System Testing:

| | |
|---|---|
| Test Case#:1 | Priority (H, L): High |
| Test Objective: Correct schedule details | |

| Test Description: Schedule is checked | | |
|---|---|---|
| Requirements Verified: Schedule is checked in the database | | |
| Test Environments: Android Emulator | | |
| Test setup: user initiates any control mechanism like schedule | | |
| Actions | Expected Results | Actual Results |
| Incorrect details | The fields get cleared and allow the user to enter correct schedule details. | In correct schedule. |
| Correct details | SMS sent at correct schedule. | SMS delivered at correct schedule. |
| Pass: Yes | Conditional pass: | Fail: |

System testing is a series of different tests whose primary purpose is to fully exercise the computer-based system. It also tests to find discrepancies between the system and its original objective, current specifications.

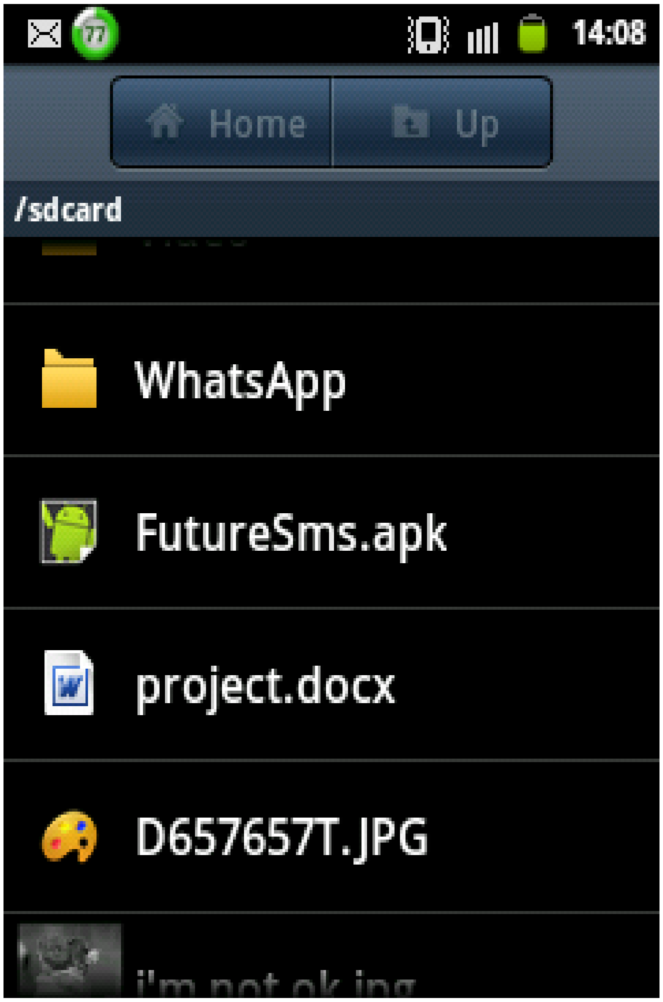
**TEST CASES**

**Test Case for Future SMS**

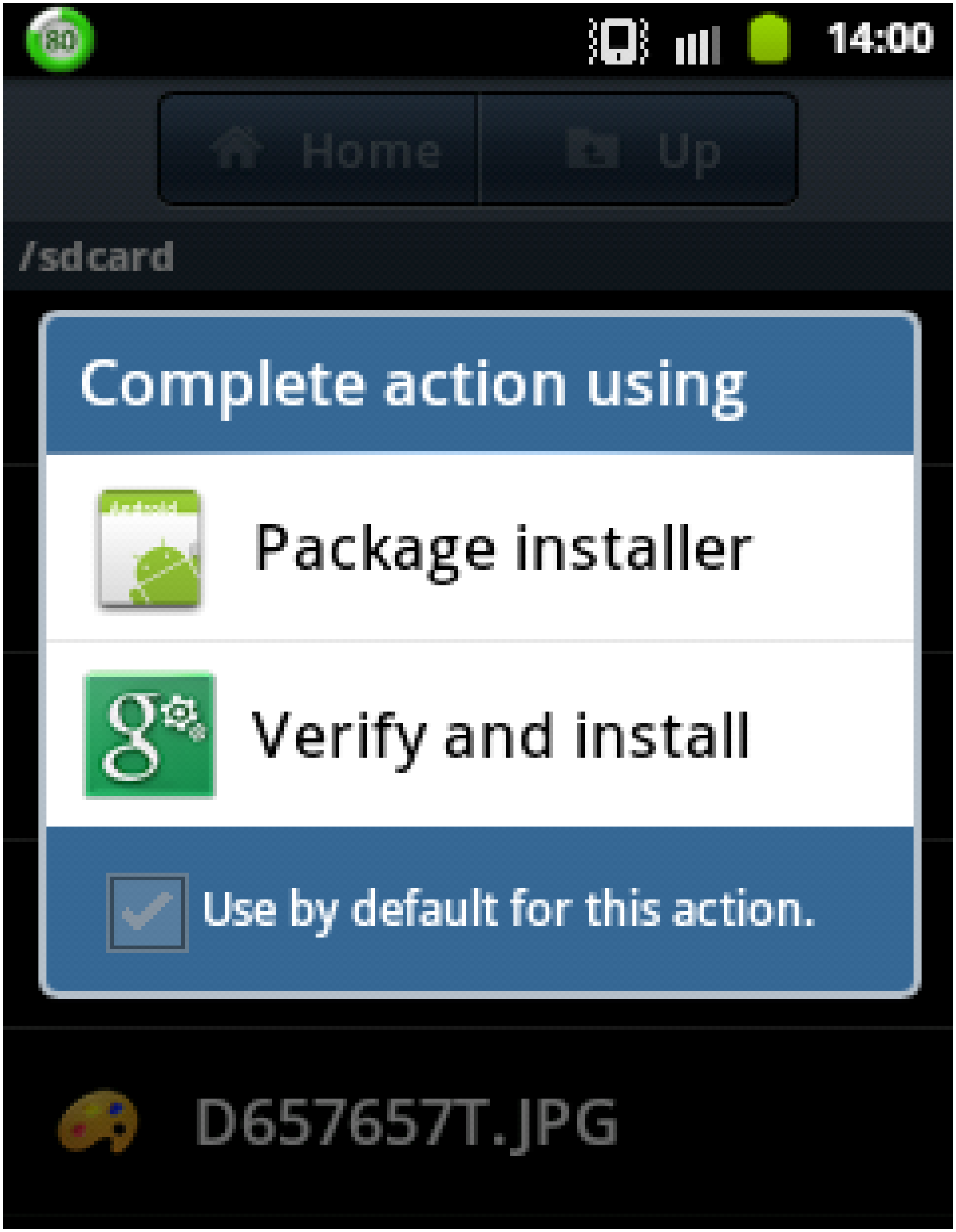
| **S. No** | **Test Case Name** | **Input** | **Expected Output** | **Actual Output** | **Result** |
|---|---|---|---|---|---|
| 1 | Title page | Click on schedule | Navigate to schedule page | Navigated to schedule page | Success |
| 2 | Entering of phone number | Enter the mobile number | Display's the mobile number | Mobile number entered | Success |
| 3 | Setting of date | Selection of date | Display's selected date | Displayed selected date | Success |
| 4 | Setting of time | Selection of time | Display's selected time | Displayed selected time | Success |
| 5 | Entering of message | Enter the required message | Display's the entered text | Displayed the entered text | Success |

| | | | | | |
|---|---|---|---|---|---|
| 6 | Navigating to title page | Click on done | Navigates to title page | Navigated | Success |
| 7 | Viewing of scheduling message | Click on done | Display's the scheduled message | Displayed the scheduled message | Success |
| 8 | Selecting of multiple phone numbers | Selecting from contacts | Displaying of multiple phone numbers | Not Displayed | Failure |

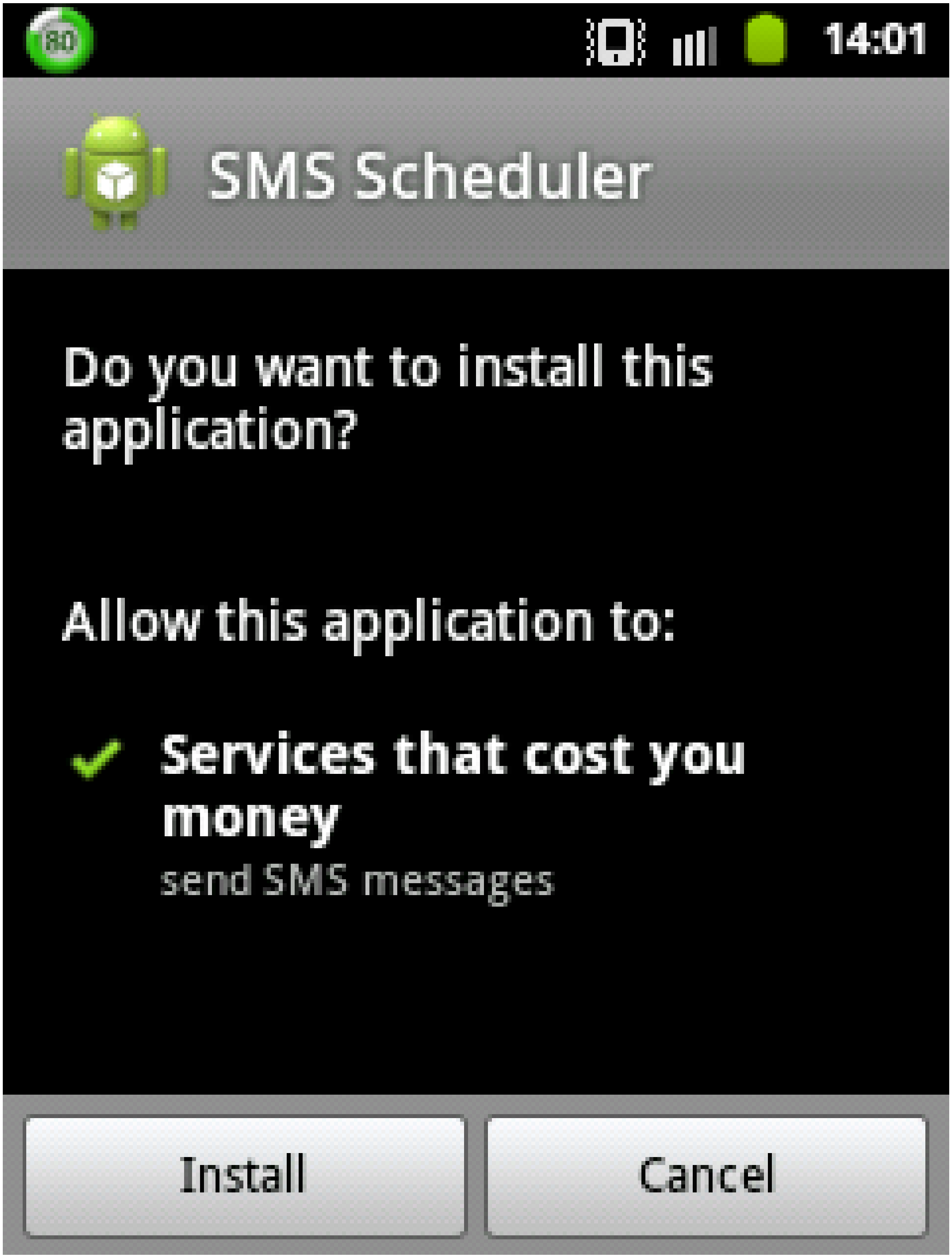
**DISCUSSION OF RESULTS**



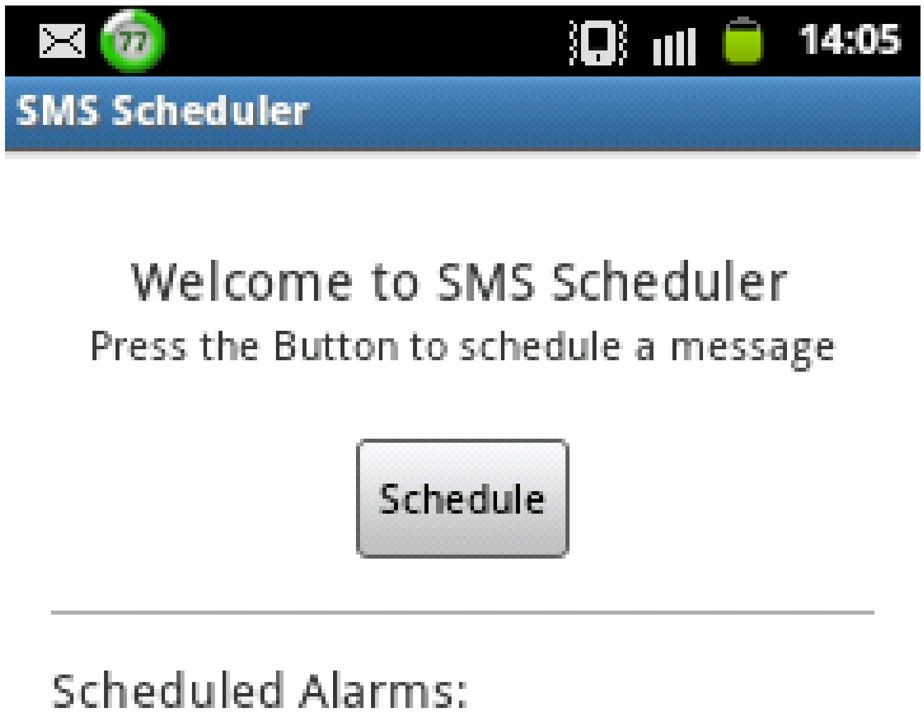

Downloaded Future SMS application. The above screen shot shows that the apk file of the Future SMS application is being downloaded.



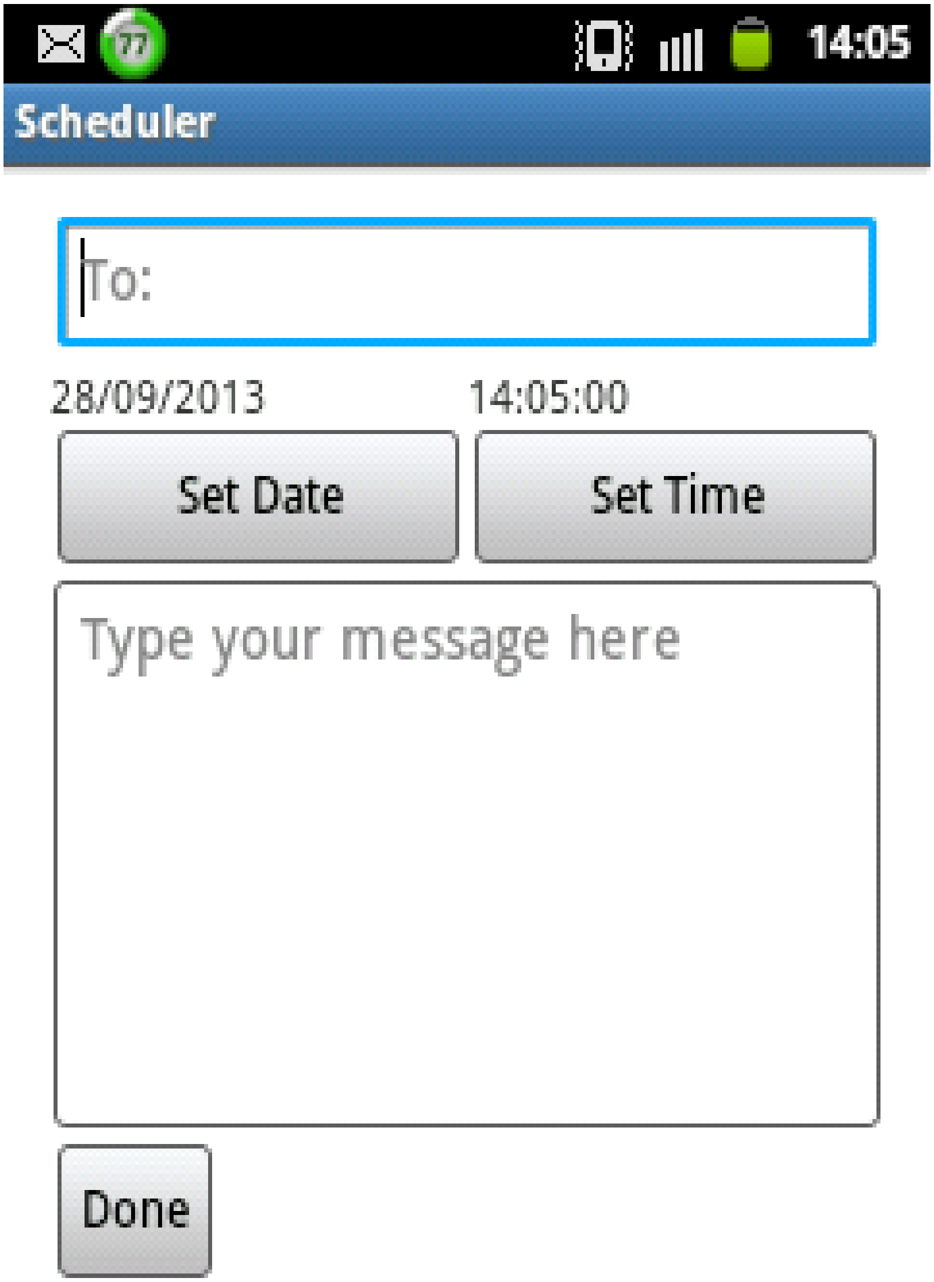

The above screen shot is to choose the action Verify and install in order to install the application.



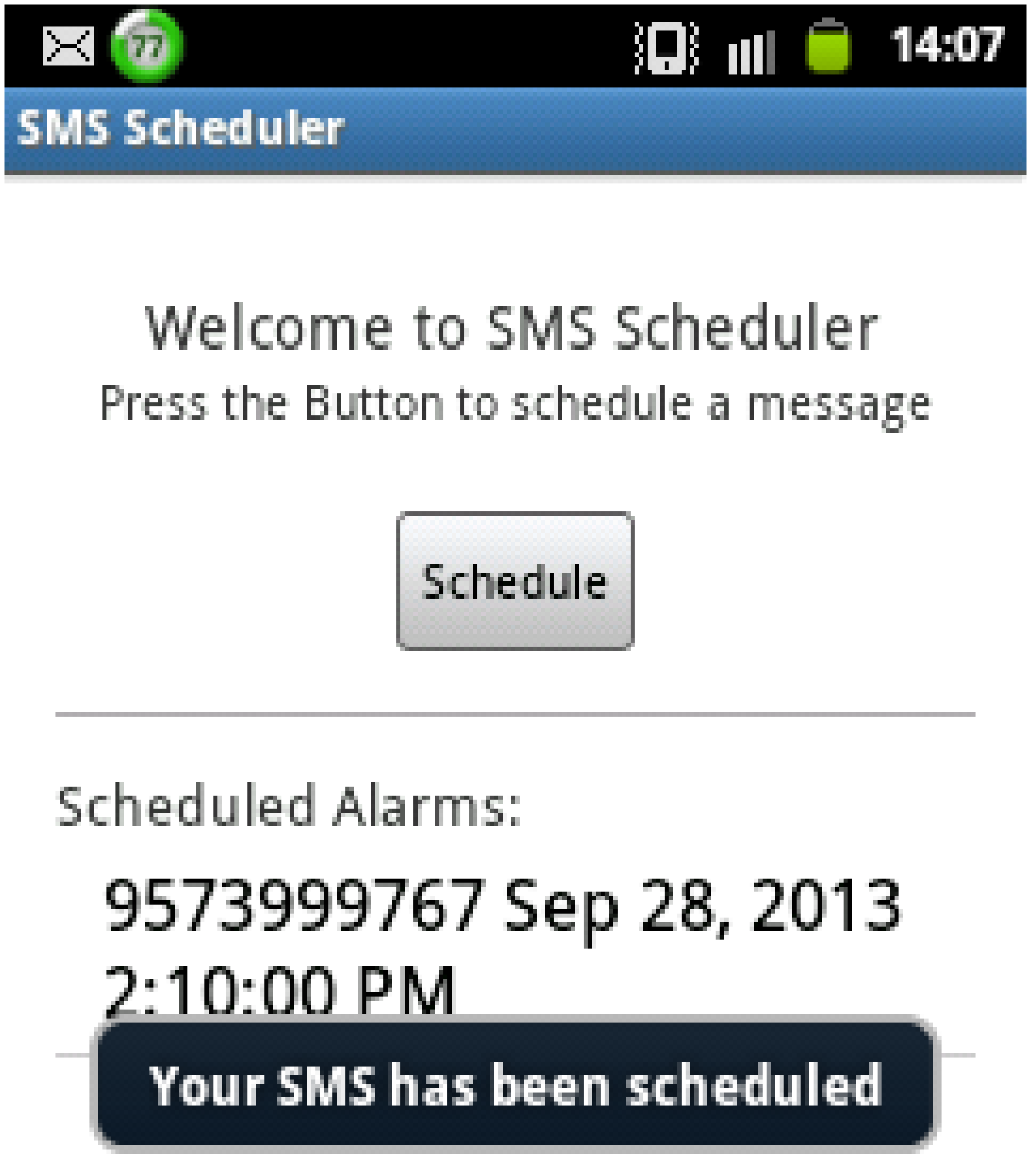

Click on the Install button in order to install the application




Click the button ‘schedule’ in order to schedule a message.




Schedule a message using SMS scheduler. Set date and time and type the required text message. Enter the recipient number and click the button ‘done’.




The above screen shows that the SMS has been scheduled. And hence the message will be sent according to the date and time scheduled.

## 8. References


- http://www.usa.gov/About/developer-resources/1usagov.shtml
- http://www.enterprisestorageforum.com/storage-management/the-impact-of-virtualized-hadoop.html
- http://searchcio.techtarget.com/opinion/Hadoop-20s-deep-impact-on-big-data-and-big-data-technologies
- http://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-hs/dependency-analysis.html
- http://www.cloudera.com/content/www/en-us/campaign/data-impact-awards.html