SENDING AND RECEIVING SMS FROM COMPUTER

By

VINAY SIDHU-021097

MAY-2006

Submitted in partial fulfillment of the Degree of Bachelor of Technology

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY-WAKNAGHAT

CERTIFICATE

This is to certify that the work entitled, "SENDING AND RECEIVING SMS FROM COMPUTER", submitted by VINAY SIDHU in partial fulfillment for the award of the degree of Bachelor of Technology in Electronics and Communication Engineering of Jaypee University of Information Technology has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Mr. Vinay Kumar

Lecturer

(Electronics and Communication Department)

ACKNOWLEDGEMENT

I wish to thank my supervisor Mr. Vinay Kumar for his generous support. Many people have contributed to this project in a variety of ways over the past few months. We also acknowledge the many helpful comments received from our teachers of the Electronics department and visualization courses and seminars. We also wish to acknowledge the University for its Services. I am indebted to Brijesh Patel, Rahul Khandelwal and Vipul Dudani who provided reviews and suggestions for improving the materials and topics covered in our package, and extend my apologies to anyone whom i have failed to mention.

Last, but not the least, I am thankful to Dr Sunil K Bhushan H.O.D (Electronics and Communication Department) for his constant reviews and suggestions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

CIMD	Computer Interface to Message Distribution
EMI	External Interface Machine
ETSI	European Telecommunication Standard Institute
GSM	Global System for Mobile Communication
GPRS	General Packet Radio System
HTTP	Hyper Text Transfer Protocol
OIS	Open Interface Specification
SIM	Subscriber Identification Module
SMSC	Short Message Service Center

ABSTRACT

The "Sending and receiving SMS from computer" facilitates a person who is working on computer can side by side receive and send text messages. The JAVA compiled files provide a user-friendly utility to the software. The distribution of the software to registered users through the website provides the distance utility. The suggestions posted in the discussion forum on the website would help in future improvement of the software.

CHAPTER 1

INTRODUCTION

1.1 SMS MESSAGING

1.1.1 What is SMS (Short Message Service)?

SMS stands for Short Message Service. It is a technology that enables the sending and receiving of messages between mobile phones. SMS first appeared in Europe in 1992. It was included in the GSM (Global System for Mobile Communications) standards right at the beginning. Later it was ported to wireless technologies like CDMA and TDMA. The GSM and SMS standards were originally developed by ETSI. ETSI is the abbreviation for European Telecommunications Standards Institute. Now the 3GPP (Third Generation Partnership Project) is responsible for the development and maintenance of the GSM and SMS standards.

As suggested by the name "Short Message Service", the data that can be held by an SMS message is very limited. One SMS message can contain at most 140 bytes (1120 bits) of data, so one SMS message can contain up to:

160 characters if 7-bit character encoding is used. (7-bit character encoding is suitable for encoding Latin characters like English alphabets.)

70 characters if 16-bit Unicode UCS2 character encoding is used. (SMS text messages containing non-Latin characters like Chinese characters should use 16-bit character encoding.)

1.1.2 SMS Messages are Supported by 100% GSM Mobile Phones

SMS messaging is a very mature technology. All GSM mobile phones support it. That not only you can exchange SMS messages with mobile users of the same wireless carrier,

but you can also exchange SMS messages with mobile users of many other wireless carriers worldwide.

1.2 What Makes SMS Messaging So Successful Worldwide?

SMS is a success all over the world. The number of SMS message exchanged every day is enormous. SMS messaging is now one of the most important revenue sources of wireless carriers. What is so special about SMS that makes it so popular worldwide? Some of the reasons are discussed below.

SMS Messages can be Sent and Read at Any Time.
SMS Messages can be sent to an Offline Mobile Phone.

1.3 Example Applications of SMS Messaging

There are many different kinds of SMS applications on the market today and many others are being developed. Applications in which SMS messaging can be utilized are virtually unlimited. We will describe some common examples of SMS applications below to give you some ideas of what can be done with SMS messaging.

Person-to-Person Text Messaging.
Alerts and Notifications.
Email, Fax and Voice Message Notifications.
E-commerce and Credit Card Transaction Alerts.

CHAPTER 2

BASIC CONCEPTS IN SMS TRANSMISSION

2.1 What is an SMS Center / SMSC?

An SMS center (SMSC) is responsible for handling the SMS operations of a wireless network. When an SMS message is sent from a mobile phone, it will reach an SMS center first. The SMS center then forwards the SMS message towards the destination. An SMS message may need to pass through more than one network entity (e.g. SMSC and SMS gateway) before reaching the destination. The main duty of an SMSC is to route SMS messages and regulate the process. If the recipient is unavailable (for example, when the mobile phone is switched off), the SMSC will store the SMS message. It will forward the SMS message when the recipient is available.

Very often, an SMSC is dedicated to handle the SMS traffic of one wireless network. A network operator usually manages its own SMSC(s) and locates them inside its wireless network system. However, it is possible for a network operator to use a third party SMSC that is located outside the wireless network system.

You must know the address of the wireless network operator's SMSC in order to use SMS messaging with your mobile phone. Typically, an SMSC address is an ordinary phone number in the international format. A mobile phone should have a menu option that can be used to configure the SMSC address. Normally, the SMSC address is pre-set in the SIM card by the wireless network operator, which means you do not need to make any change to it.

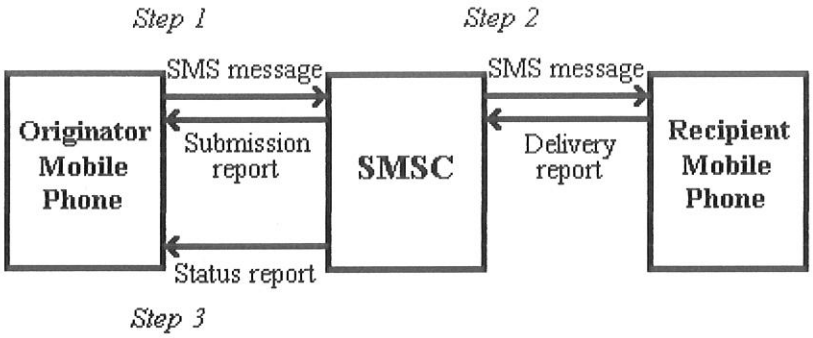

Fig 1 the following figure illustrates the transmission process of an SMS Message.

2.2 What is an SMS Gateway?

One problem of SMS messaging is that SMSCs developed by different companies use their own communication protocol and most of these protocols are proprietary. For example, Nokia has an SMSC protocol called CIMD whereas another SMSC vendor, CMG, has an SMSC protocol called EMI. We cannot connect two SMSCs if they do not support a common SMSC protocol. To deal with this problem, an SMS gateway is placed between two SMSCs. This is illustrated in the following figure. The SMS gateway acts as a relay between the two SMSCs. It translates one SMSC protocol to another one. This way can be used by two different wireless carriers to interconnect their SMSCs for purposes such as enabling the exchange of inter-operator SMS messages.

Figure 2. An SMS gateway acts as a relay between two SMS centers.

2.2.1 Problem of following different protocols by different SMS centers.

To deal with this problem, an SMS gateway can be set up to handle the connections to the SMSCs. Now the SMS text messaging application only needs to know how to connect to the SMS gateway. To support more SMSCs, you just need to modify the settings of the SMS gateway. No change to the source code of the SMS text messaging application is required. The use of an SMS gateway can greatly shorten the SMS text messaging application's development time.

To connect to an SMS gateway, you can use an SMSC protocol such as SMPP and CIMD. Some SMS gateways support an HTTP / HTTPS interface. HTTP / HTTPS is easier to use than SMSC protocols. The drawback is that there may be fewer SMS features to use. For example, an SMS gateway may not support the sending of picture messages through the HTTP interface

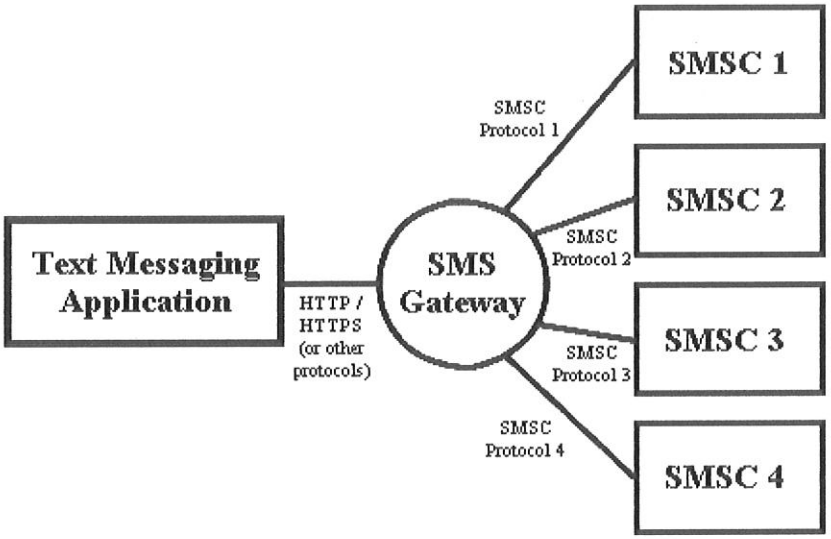

Figure 3. An SMS text messaging application connects to SMSCs through an SMS gateway

CHAPTER 3

HOW TO SEND SMS MESSAGES FROM COMPUTER

There are two ways to send SMS messages from a computer / PC to a Mobile phone.

Connect a mobile phone or GSM/GPRS modem to a computer / PC. Then use the computer / PC and AT commands to instruct the mobile phone or GSM/GPRS modem to send SMS messages

Connect the computer / PC to the SMS center (SMSC) or SMS gateway of a wireless carrier or SMS service provider. Then send SMS messages using a protocol / interface supported by the SMSC or SMS gateway

3.1.1 The First Way: Sending SMS Messages from a Computer Using a Mobile Phone or GSM/GPRS Modem

The SMS specification has defined a way for a computer to send SMS messages through a mobile phone or GSM/GPRS modem. A GSM/GPRS modem is a wireless modem that works with GSM/GPRS wireless networks. A wireless modem is similar to a dial-up modem. The main difference is that a wireless modem transmits data through a wireless network whereas a dial-up modem transmits data through a copper telephone line. Most mobile phones can be used as a wireless modem. However, some mobile phones have certain limitations comparing to GSM/GPRS modems.

To send SMS messages, first place a valid SIM card from a wireless carrier into a mobile phone or GSM/GPRS modem, which is then connected to a computer. There are several ways to connect a mobile phone or GSM/GPRS modem to a computer. For example, they can be connected through a serial cable, a USB cable, a Bluetooth link or an infrared link. The actual way to use depends on the capability of the mobile phone or GSM/GPRS modem. For example, if a mobile phone does not support Bluetooth, it cannot connect to the computer through a Bluetooth link.

After connecting a mobile phone or GSM/GPRS modem to a computer, you can control the mobile phone or GSM/GPRS modem by sending instructions to it. The instructions used for controlling the mobile phone or GSM/GPRS modem are called AT commands. (AT commands are also used to control dial-up modems for wired telephone system.) Dial-up modems, mobile phones and GSM/GPRS modems support a common set of standard AT commands. In addition to this common set of standard AT commands, mobile phones and GSM/GPRS modems support an extended set of AT commands. One use of the extended AT commands is to control the sending and receiving of SMS messages.

The following table lists the AT commands that are related to the writing and sending of SMS messages:

SMSC vendor	SMSC protocol
CMG (CMG and Logical have merged into LogicaCMG.)	EMI (External Machine Interface) UCP (Universal Computer Protocol)
Logica (CMG and Logica have merged into LogicaCMG.) (Now the SMS Forum is responsible for the development of SMPP.)	SMPP (Short Message Peer to Peer)
Nokia	CIMD (Computer Interface to Message Distribution)
SEMA Group (Now Airwide Solutions)	OIS (Open Interface Specification) SMS200

One way to send AT commands to a mobile phone or GSM/GPRS modem is to use a terminal program. A terminal program's function is like this: It sends the characters you typed to the mobile phone or GSM/GPRS modem. It then displays the response it receives from the mobile phone or GSM/GPRS modem on the screen. The terminal program on Microsoft Windows is called HyperTerminal.

To send SMS messages from an application, one has to write the source code for connecting to and sending AT commands to the mobile phone or GSM/GPRS modem, just like what a terminal program does. You can write the source code in C, C++, Java, Visual Basic, Delphi or other programming languages you like.

3.1.2 The Second Way

The first way for sending SMS messages from a computer through a mobile phone or GSM/GPRS modem has a major limitation that is the SMS sending rate is too low. If one need a high SMS sending rate, obtaining a direct connection to the SMS center (SMSC) or SMS gateway of a wireless carrier is necessary. The connection may be made through the Internet, X.25 or dial-up. If you cannot get a direct connection to the SMSC or SMS gateway of a wireless carrier, another choice is to get a connection to the SMS gateway of an SMS service provider, which will forward SMS messages towards a suitable SMSC.

CHAPTER 4

HOW TO RECIEVE SMS MESSAGES USING COMPUTER

4.1 In general, there are two ways to receive SMS messages using computer / PC

Connect a mobile phone or GSM/GPRS modem to a computer / PC. Then use the computer / PC and AT commands to get the received SMS messages from the mobile phone or GSM/GPRS modem.

Get access to the SMS center (SMSC) or SMS gateway of a wireless carrier. Any SMS message received will be forwarded to your computer / PC using a protocol / interface supported by the SMSC or SMS gateway

4.1.1 The First Way

Receiving SMS messages through a mobile phone or GSM/GPRS modem has a major advantage over the other two ways -- wireless carriers usually do not charge any fee for receiving incoming SMS messages with their SIM cards. The disadvantage of receiving SMS messages this way is that a mobile phone or GSM/GPRS modem cannot handle a large amount of SMS traffic. One way to overcome this is to load balance the SMS traffic with a pool of mobile phones or GSM/GPRS modems. Each mobile phone or GSM/GPRS modem will have its own SIM card and mobile phone number.

In terms of programming, sending and receiving SMS messages through a mobile phone or GSM/GPRS modem are similar. What you need to do is to send instructions (in the form of AT commands) to the mobile phone or GSM/GPRS modem.

The following table lists the AT commands that are related to the receiving and reading of SMS messages:

AT command	Meaning
+CNMI	New message indications
+CMGL	List messages
+CMGR	Read messages
+CNMA	New message acknowledgement

4.1.2 The Second Way

Like sending SMS messages, receiving SMS messages through a mobile phone or GSM/GPRS modem has a major limitation that is the SMS transmission rate is too low. If you need a high SMS transmission rate, one way is to obtain a direct connection to the SMS center (SMSC) or SMS gateway of a wireless carrier. The connection may be made through the Internet, X.25 or dial-up.

To receive SMS messages through a direct connection to the SMSC or SMS gateway of a wireless carrier, usually you need to contact the wireless carrier staff to discuss the details. A wireless carrier may only provide such service to those who have huge SMS traffic and the total fee can be very high. (However, if you divide the total fee by the number of SMS messages that is allowed to receive through the SMSC or SMS gateway, you may find that the fee per incoming SMS message is very low.) The service may involve a one-time setup fee, a monthly subscription fee and/or a per-message fee. To know the exact cost of the service, protocols supported, network coverage and other information, usually you have to contact the wireless carrier staff.

4.2 AT Commands

This is intended to show the differences of AT+C command set implementation of various GSM devices. Other than ETSI, standardized commands are not listed here although there are tons of manufacturer specific commands in many implementations. I repeat: some devices implements the same or similar functions listed below but with their own, non-standard commands.

// read some information from the mobile phone

//----------

//----- get attention

at

ok

//----- get signal quality

at+csq

+CSQ: 31,99

//----- get battery charge

at+cbc

+CBC: 0,90

//----- check if PIN is verified

at+cpin?

+CPIN: READY

//----- check network registration

at+creg?

+CREG: 0,1

//----- request model identification

at+cgmm

Nokia 6210

//----- request model identification

at+cgmm

SL55

//----- request international mobile subscriber identity

at+cimi

262017130021182

//----- get message format

at+cmgf=?

+CMGF: (0)

//----- get phone activity status

at+cpas

+CPAS: 0

//--

// list and delete SMS in mobile phone

//--

//----- 1 SMS is stored (index=2) and could be read with "+CMGL"

at+cmgl

+CMGL: 2,1,,43

0791947101670000040485080039004010313131604020D737DB7C0EBBCF2E69D8B

D6603C865D739

DD22975DE3771B442DCFE9

OK

//----- 2 SMS are stored (index=1, 2)and could be read with "+CMGL"

at+cmgl

+CMGL: 1,0,,51

0791947101670000040485080039004010411171334029D737DB7C0EBBCF2E69D8B

D6603C865D739

DD22975DE3771B747DB3CDE7B0FB0CA296E774

+CMGL: 2,1,,43

07919471016700000404850800390040103131316040 20D737DB7C0EBBCF2E69D8B

D6603C865D739

DD22975DE3771B442DCFE9

OK

//----- 1 SMS (index=1) will be deleted by "+CMGD=1"

at+cmgd=1

OK

//----- 1 SMS is stored (index=2) and could be read with "+CMGL"

at+cmgl

+CMGL: 2,1,,43

07919471016700000404850800390040103131316040 20D737DB7C0EBBCF2E69D8B

D6603C865D739

DD22975DE3771B442DCFE9

OK

//----- 1 SMS (index=2) will be deleted by "+CMGD=2"

at+cmgd=2

OK

//----- 0 SMS are stored and could be read with "+CMGL"

at+cmgl

OK

CHAPTER 5

CODE

SMS.java

```
package smspack;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.text.*;
import javax.swing.*;

public class SMS extends JFrame implements ActionListener {
// TOA = type of address (= type of number || numbering plan identifier)
// it is a national dialing number format (0x81, e.g. short number format for net internal calls)
static final boolean TOA_NATIONAL = true;
// it is a international dialing number format (
static final boolean TOA_INTERNATIONAL = false;
static final String SMSTEXTSIGN = ">";          // sign for the begin of the SMS text
static final int TIMEOUT = 2*60*1000;           // timeout [ms] for SMS run time measurement
static final int MAXNOSIMSMS = 20;              // maximal number of stored SMS in SIM
static final String SMSDIALNO = "0175108xxxx";  // dialing number for SMS transmission, this is the number of the phone you want to send the SMS
static final String EMAILDIALNO = "8000";       // dialing number for eMail transmission, this is the number you have to dial if you want to convert a SMS into a e-Mail
static final String EMAILADR = "vinay.sidhu@gmal.com"; // eMail adress
static final String COMPORT = "COM1";           // used COM port for data transmission
```

```
public static JTextArea ta = new JTextArea(40, 100);

SMS() {
//----- create text area with scroll pane
ta.setTabSize(4);
ta.setFont(new Font("Monospaced", 0, 12));     // Courier, 12 point size
ta.setBackground(new Color(  0,   0,   0));    // black blue
ta.setForeground(new Color(200, 255, 100));    // yellowgreen
getContentPane().add(new JScrollPane(ta), BorderLayout.CENTER);
//----- create panel with buttons
JPanel buttonPanel = new JPanel(new GridLayout(25, 1));
JButton button1 = new JButton("decode PDUs");
button1.addActionListener(this);
buttonPanel.add(button1);
JButton button8 = new JButton("create PDUs");
button8.addActionListener(this);
buttonPanel.add(button8);
JButton button2 = new JButton("read SMS");
button2.addActionListener(this);
buttonPanel.add(button2);
JButton button3 = new JButton("read ME Info");
button3.addActionListener(this);
buttonPanel.add(button3);
JButton button4 = new JButton("send SMS");
button4.addActionListener(this);
buttonPanel.add(button4);
JButton button5 = new JButton("invest Runtime");
button5.addActionListener(this);
buttonPanel.add(button5);
JButton button6 = new JButton("AT Cmd.");
button6.addActionListener(this);
```

```
buttonPanel.add(button6);
JButton button7 = new JButton("Parse");
button7.addActionListener(this);
buttonPanel.add(button7);
JButton button9 = new JButton("Exit");
button9.addActionListener(this);
buttonPanel.add(button9);
getContentPane().add(buttonPanel, BorderLayout.EAST);
} // SMS

/** creates a Windows conform formated date and time string
*  @return: data and time string
*/
public static String getFormatedDateTime() {
String s;
Date t = new Date();
t.getTime();
s = DateFormat.getDateTimeInstance().format(t);
return s;
} // getFormatedDateTime

/** creates a long value for the time
*  @return: number of milliseconds since January 1, 1970,
*/
public static long getLongTime() {
long l;
Date time = new Date();
l = time.getTime();
return l;
} // getLongTime
```

```
/** show text in the text window
*  @param text text string to show on the display
*/
public static void showText(String text) {
ta.setText(ta.getText() + text);
} // showText


/** show text in the text window
*  @parameter text char array to show on the display
*/
public static void showText(char[] text) {

ta.setText(ta.getText() + text);
}
/** dispatcher which the main parts of the programm
*  @param event event button which is pressed
*/
public void actionPerformed(ActionEvent event) {
boolean loopbreak;
int n, i;
int[] index;
long sendtime, starttime, receivetime, deltatime;
String s, pdu, sendtext, receivetext, originateno;

showText("\n----------------------------------------\n");
String cmd = event.getActionCommand();
if (cmd.equals("decode PDUs")) {
showText("decode some PDUs for demonstration and test\n\n");
pdu = "0791";
```

```
showText(SMSTools.getSMSText(pdu)+ "\n");
pdu = "0791";
showText(SMSTools.getSMSText(pdu)+ "\n");
pdu = "0791 ";
showText(SMSTools.getSMSText(pdu) + "\n");
pdu = "0791";
showText(SMSTools.getSMSText(pdu)+ "\n");
pdu = "0791";
showText(SMSTools.getSMSText(pdu)+ "\n");
} // if

else if (cmd.equals("read SMS")) {
showText("read and decode all SMS from SIM\n\n");
try {
Port.open();
if (Msg.test() == false) {
showText("ERROR: Can't connect to mobile phone\n");
} // if
} // try
catch (Exception e) {showText("Error: Exception in read SIM: " + e);}

try {
n = Msg.getNoOfSMS();
showText("No. of stored SMS in SIM: " + Integer.toString(n) + "\n");
index = Msg.getIndexOfSMS();        // get a index list of stored SMS
n = -1;
do {
n++;
if (index[n] != 0) {        // found a guilty index for a SMS
i = index[n];
s = Msg.getSMS(i);
```

```
s = SMSTools.getSMSText(s);
showText("SMS No. " + index[n] + ": " + s + "\n");
} // if
} while (index[n] != 0);
Port.close();
} // try
catch (Exception e) {
showText("Exception in Main: " + e);
} // catch
} // else if

else if (cmd.equals("read ME Info")) {
showText("Get some information about the connected mobile phone\n\n");
try {
Port.open();
showText("Battery Status:      " + Msg.getBatteryStatus() + "\n"); // get battery status
from ME
showText("Signal Quality:      " + Msg.getSignalQuality() + "\n"); // get signal quality
from ME
Port.close();
} // try
catch (Exception e) {
showText("Error: Exception in read ME Info: " + e);
} // catch
} // else if

else if (cmd.equals("send SMS")) {
showText("send some SMS and eMails\n\n");
try {
Port.open();
```

```
Msg.sendSMS(EMAILDIALNO, TOA_NATIONAL, EMAILADR + "Testmail, PC send
time: " + getFormatedDateTime());
Msg.sendSMS(SMSDIALNO, TOA_INTERNATIONAL, "Test-SMS, PC send time: " +
getFormatedDateTime());
Msg.sendSMS(EMAILDIALNO, TOA_NATIONAL, EMAILADR + " Testmail");
Port.close();
} // try
catch (Exception e) {
showText("Exception in Main: " + e);
} // catch
} // else if

else if (cmd.equals("invest Runtime")) {
showText("Investigate run time of a SMS within the GSM network\n\n");

try {
Port.open();
showText("Delete all SMS from SIM\n");
Msg.deleteAllSMS();
sendtime = getLongTime();
sendtext = "Testmail for run time, PC send time: " + getFormatedDateTime();
receivetext ="";
loopbreak = false;
Msg.sendSMS (SMSDIALNO, TOA_NATIONAL, sendtext);   // send a SMS to the
sending mobile phone
do {
Thread.sleep(500);                    // wait 500 msec
receivetime = getLongTime();
deltatime = (receivetime - sendtime)/1000;   // deltatime is in seconds
index = Msg.getIndexOfSMS();              // get a index list of stored SMS
n = -1;
```

```
do {
n++;
if (index[n] != 0) {                    // found a guilty index for a SMS
receivetext = SMSTools.getSMSText(Msg.getSMS(index[n]));
i = receivetext.indexOf(SMSTEXTSIGN);
receivetext = receivetext.substring(i+1, receivetext.length());
if (sendtext.compareTo(receivetext) == 0) {
loopbreak = true;                       // received the same SMS as sended
break;
} // if
} // if
} while (index[n] != 0);
if (deltatime >= TIMEOUT) loopbreak = true;
} while (loopbreak == false);
if (deltatime >= TIMEOUT) showText("timeout of " + TIMEOUT + " s reached\n");
else  showText("actual run time of a SMS = " + deltatime + " s\n");
Port.close();
}  // try
catch (Exception e) {
showText("Exception in Main: " + e);
}  // catch
} // else if

else if (cmd.equals("AT Cmd.")) {
showText("send some AT commands and display the received raw data\n\n");
try {
Port.open();
showText("Mobile Phone Modell: " + Port.sendAT ("AT+CGMM"));  // get
Modellinformation from ME
showText("Connection Status:   " + Port.sendAT ("AT+CREG?")); // get connection
status to GSM net from ME
```

```
Port.close();
} // try
catch (Exception e) {
showText("Error: Exception in AT Cmd.: " + e);
} // catch
} // else if

else if (cmd.equals("Parse")) {
showText("parse received SMS, parsing is switched automatically off in " +
TIMEOUT/1000 + " s\n\n");
loopbreak = false;
starttime = getLongTime();
try {
Port.open();
Msg.deleteAllSMS();
do {
showText("Delete all SMS from SIM\n");
index = Msg.getIndexOfSMS();              // get a index list of stored SMS
n = -1;
do {
n++;
if (index[n] != 0) {                      // found a guilty index for a SMS
receivetext = SMSTools.getSMSText(Msg.getSMS(index[n]));
i = receivetext.indexOf(" ");
originateno = receivetext.substring(0, i);
i = receivetext.indexOf(SMSTEXTSIGN);
receivetext = receivetext.substring(i+1, receivetext.length());
showText("received SMS from: " + originateno + ", content: " + receivetext + "\n");
if (receivetext.indexOf(".on") == 0) {
showText("found command in SMS: .on\n");
} // if
```

```
else if (receivetext.indexOf(".off") == 0) {
showText("found command in SMS: .off\n");
} // else if
else if (receivetext.indexOf("get time") == 0) {
showText("found command in SMS: get time\n");
showText("send local time to the sender\n");
Msg.sendSMS(SMSDIALNO, TOA_INTERNATIONAL, "PC send time: " +
getFormatedDateTime());
} // else if
else if (receivetext.indexOf("exit") == 0) {
showText("found command in SMS: exit\n");
loopbreak = true;
} // else if
else {    // unknown command
showText("no command found in SMS\n");
showText("send help information to the sender\n");
Msg.sendSMS(SMSDIALNO, TOA_INTERNATIONAL, "possible commands: .on, .off,
get time, exit");
} // else
} // if
} while (index[n] != 0);
Msg.deleteAllSMS();
Thread.sleep(1000);                    // wait 1 sec
showText("time: " + getFormatedDateTime() + "\n\n");
deltatime = getLongTime() - starttime;          // deltatime is in milliseconds
if (deltatime >= TIMEOUT) loopbreak = true;
} while (loopbreak == false);
if (deltatime >= TIMEOUT) showText("timeout of " + TIMEOUT/1000 + " s reached");
Port.close();
}  // try
catch (Exception e) {
```

```
showText("Error: Exception in parsing SMS: " + e);
} // catch
} // else if

else if (cmd.equals("create PDUs")) {
showText("creates PDUs for sending PDUs from a simple microcontroller to the mobile
phone\n\n");
String dialno4SMS, dialno4Email, emailadr, message = new String();
dialno4SMS   = "01751082323";   // dialing number for SMS transmission, this is the
number of the phone you want to send the SMS
dialno4Email = "8000";          // dialing number for eMail transmission, this is the
number you have to dial if you want to convert a SMS into a e-Mail
emailadr     = "wolfgang.rankl@de.gi-de.de";  // eMail adress
message      = "et's geht's des glump";       // message to the receiver, maximal length is
160 characters

SMS.showText("Dialing Number for SMS:   " + dialno4SMS   + "\n");
SMS.showText("Dialing Number for EMail: " + dialno4Email + "\n");
SMS.showText("Email Adress:             " + emailadr     + "\n");
SMS.showText("Message:                  " + message      + "\n");

SMSTools smstools = new SMSTools();
byte[] pdu_byte = SMSTools.getPDUPart(dialno4SMS, TOA_INTERNATIONAL,
message); // build PDU for SMS
char[] pdu_char = SMSTools.toHexString(pdu_byte);
pdu = SMSTools.convertCharArray2String(pdu_char);
SMS.showText("PDU for SMS:              " + pdu          + "\n");

pdu_byte = SMSTools.getPDUPart(dialno4Email, TOA_NATIONAL, message);    //
build PDU for eMail
pdu_char = SMSTools.toHexString(pdu_byte);
```

```
pdu = SMSTools.convertCharArray2String(pdu_char);
SMS.showText("PDU for EMail:          " + pdu        + "\n");
} // else if

else if (cmd.equals("Exit")) {
showText("\tExit Button pressed\n");
showText("Stop Program GUI\n");
System.exit(0);
} // else if
showText("-----------------------------------------\n");
} // actionPerformed

/** main
*  @param args there will be no arguments for the main
*/
public static void main(String [] args){
//--- build the GUI
SMS sms = new SMS();
sms.setLocation(0, 0);
sms.pack();
sms.setTitle("SMS Transceiver");
sms.setVisible(true);
showText("Start Program SMS Transceiver\n\n");
} // main
} // class
```

CHAPTER 6

FUTURE SCOPE

The work "sending and receiving SMS from computer" has very bright prospects in future. The topic coverage of this work can be extended to services like we have rediffbol ,Google talk etc.

SMS Transceiver can be used for applications like to send mass emails with a daily horoscope (10 mobile phones are used in parallel on one PC with SMS Transceiver), remote control of heating (cottage in the alps), daily transfer of sales from kiosk, regular temperature transmission, alarm systems and many applications for data transmission of automated measured values.

CHAPTER 7

TOOLS

7.1 JAVA

Java is an Object Oriented language based on C. Many years in gestation, it has shown an explosive growth in popularity since its release. This is partly because of the relative simplicity of the language when compared to C and particularly languages such as C++, but also because the design and implementation of Java allow compiled programs to be shipped across the Internet to run on client systems.

This network capability was shown in the HotJava World Wide Web browser, which would accept an extended form of HTML containing compiled Java programs (applets) and run those programs locally. This has since been adopted by other browsers such as Netscape. HotJava was, of course, entirely written in Java using the Java libraries.

Since then Java has been licensed to Borland and Microsoft among others, and the growth looks set to continue.

Why Java is of particular interest to readers of the X Advisor is in the AWT (Abstract Window Types) library that is used for the GUI components of HotJava and that can also be used for the GUI components of applets. The library is complete enough to both build fully functional standalone GUI applications, to perform drawings and show images and animations.

CONCLUSION

The "Sending and receiving SMS from computer" has been developed to send and receive SMS while one is working on the computer with so much ease. Since ,there is no problem regarding the connection so it is more reliable mode of communication. Even if the recipient mobile is switched off, SMS can be send offline and as soon as he switch on the mobile, he will get all those messages.

Each topic is dealt with extreme thoroughness . The code that is written in JAVA is extremely simple and easy to understand. The distribution of the code to registered users through the website provides the distance utility. The suggestions posted in the discussion forum on the website would help in future improvement of the software.

.

BIBLIOGRAPHY

http://www.uakron.edu/library/instruction/glossary.htm
http://www.centc251.org/Ginfo/Glossary/tcglosu.htm
http://www.scotsmist.co.uk/glossary_u.html