SKRIPSI

PENGEMBANGAN APLIKASI TEXT TO SPEECH BERBAHASA INDONESIA SEBAGAI PEMBACA SMS MENGGUNAKAN BORLAND DELPHI 7

Disusun Oleh :
ROZEIN ROUSYANFIKRI
NIM 05.12.577

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2009

LEMBAR PERSETUJUAN

PENGEMBANGAN APLIKASI TEXT TO SPEECH BERBAHASA INDONESIA SEBAGAI PEMBACA SMS MENGGUNAKAN BORLAND DELPHI 7

SKRIPSI

Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh
Gelar Sarjana Teknik Komputer Dan Informatika Strata Satu (S–1)

Disusun Oleh :
ROZEIN ROUSYANFIKRI
05.12.577

Diperiksa dan Disetujui

Mengetahui Ketua Jurusan Teknik Elektro S-1	Dosen Pembimbing
Ir. F. Yudi Limpraptono, MT	**Ir. F. Yudi Limpraptono, MT**
NIP Y. 1039500274	**NIP Y. 1039500274**

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG

2009

PENGEMBANGAN APLIKASI TEXT TO SPEECH BERBAHASA INDONESIA SEBAGAI PEMBACA SMS MENGGUNAKAN BORLAND DELPHI 7

Rozein Rousyanfikri
Institut Teknologi Nasional Malang
Di Bawah Bimbingan
Bapak Ir. F. Yudi Limpraptono, MT.

ABSTRAK

Perkembangan teknologi komunikasi yang semakin pesat dari waktu ke waktu memudahkan seseorang untuk mendapatkan informasi tanpa dibatasi oleh ruang dan waktu. Salah satu hasil dari perkembangan teknologi komunikasi yaitu ponsel (telephone seluler). SMS (Short Message Service) merupakan bagian dari fasilitas ponsel. SMS menyediakan layanan komunikasi berupa pengirim dan penerima informasi dalam bentuk pesan singkat. Fasilitas SMS ini tidak akan bermanfaat secara maksimal apabila kemampuan dari penggunanya mengalami kesulitan membaca seperti orang yang menderita tuna aksara dan tuna netra. Oleh sebab itu pada penelitian ini akan dibuat suatu aplikasi sebagai pembaca SMS dengan menggunakan Speech Engine Embrola yang dibuat oleh DR. Arry Akhmad Arman sebagai pembangkit ucapannya yang telah dikembangkan oleh team kioss project (by Lury Darmawan) menjadi Tukang Omong PD.

Dengan adanya aplikasi pembaca SMS ini diharapkan bisa membantu dan memudahkan penyandang tuna nentra dan tuna aksara dalam memperoleh informasi berupa SMS, pada penulisan ini juga akan dijelaskan tentang tahap pengerjaan, mulai dari proses analisa, perencanaan dan konstruksi yang menggunakan aplikasi Borland Delphi 7, Microsoft office Access 2007 dan pembangkit ucapan berupa aplikasi Speech Engine Embrola (Tukang Omong PD), setelah aplikasi memparsing data PDU(Protokol Data Unit) dan mendapatkan string dari isi SMS maka aplikasi akan mengirimkan string tersebut ke aplikasi pembangkit ucapan untuk nantinya di ucapkan.

Kata Kunci : TTS (text to Speech), Pembaca SMS.

KATA PENGANTAR

Dengan mengucap puji syukur kehadirat Allah SWT yang dengan segala Kasih dan Anugerah - Nya, telah memberikan kekuatan, kesabaran, bimbingan dan perlindungan sehingga penulis dapat menyelesaikan laporan skripsi dengan judul: " **PENGEMBANGAN APLIKASI TEXT TO SPEECH BERBAHASA INDONESIA SEBAGAI PEMBACA SMS MENGGUNAKAN BORLAND DELPHI 7** " . Pembuatan skripsi ini disusun guna memenuhi syarat akhir kelulusan pendidikan jenjang Strata - 1 di Institut Teknologi Nasional Malang. Dalam penyusunan skripsi ini penulis banyak mendapat bantuan baik moril maupun materiil, saran dan dorongan semangat dari berbagai pihak, untuk itu penulis mengucapkan terima kasih kepada :

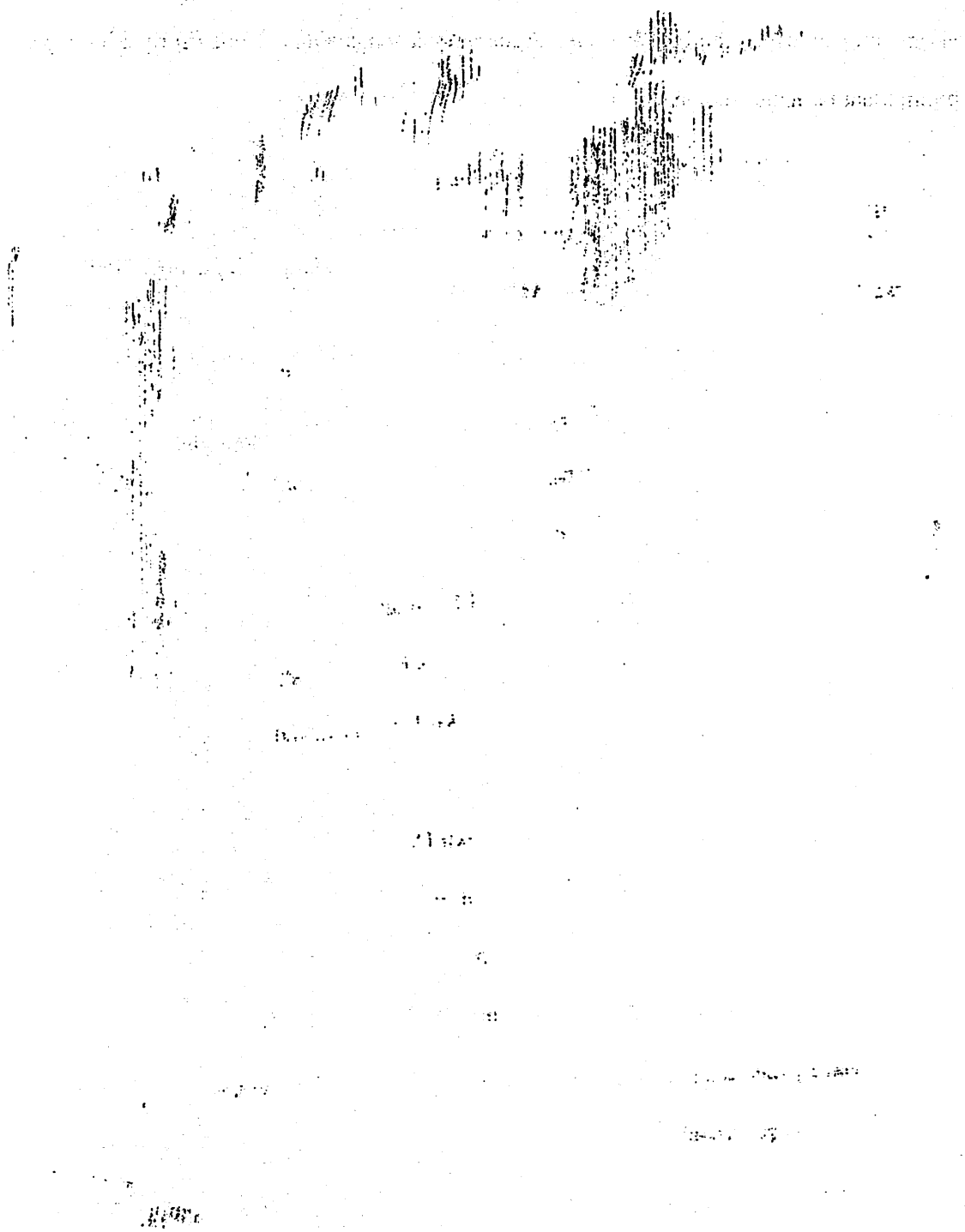
1. Bapak Prof. Dr. Ir. Abraham Lomi, MSEE., selaku Rektor ITN Malang.
2. Bapak Ir. H. Sidik Noertjahyono, MT., selaku Dekan Fakultas Teknologi Industri.
3. Bapak Ir. F. Yudi Limpraptono, MT., selaku Ketua Jurusan Teknik Elektro S – 1 ITN Malang
4. Bapak Ir. F. Yudi Limpraptono, MT., selaku Dosen Pembimbing.
5. Keluarga besar yang selalu mendukung dan mendoakan saya.
6. Semua pihak yang telah membantu penulis dalam menyelesaikan skripsi ini yang tidak bisa penulis sebutkan satu persatu.

Penulis menyadari bahwa laporan ini masih banyak yang perlu disempurnakan. Oleh sebab itu kritik dan saran yang membangun sangat diharapkan.

Akhir kata, penulis mohon maaf kepada semua pihak bilamana selama penyusunan skripsi ini penyusun membuat kesalahan secara tidak sengaja dan semoga skripsi ini dapat bermanfaat bagi kita semua, Amin.

Malang, September 2009

Penulis



DAFTAR ISI

LEMBAR PERSETUJUAN ..... **i**

ABSTRAK ..... **ii**

KATA PENGANTAR ..... **iii**

DAFTAR ISI ..... **v**

DAFTAR GAMBAR ..... **viii**

DAFTAR TABEL ..... **x**

BAB I PENDAHULUAN **1**

1.1. Latar Belakang 1

1.2. Rumusan Masalah 2

1.3. Batasan Masalah 2

1.4. Tujuan Penelitian 3

1.5. Metode Penelitian 4

1.6. Sistematika Penulisan 5

BAB II LANDASAN TEORI **6**

2.1. SMS (Short Message Service) 6

2.1.1. Struktur Jaringan SMS 8

2.1.2. Pengiriman dan Penerimaan SMS 11

2.1.2.1. PDU untuk mengirim SMS ke SMSC 14

2.1.2.2. PDU SMS yang diterima dari SMSC 20

2.2. Perintah AT (AT Command) 24

2.3. Bagan Alir Program (Flowchart) 26

2.4. TTS (Text To Speech) 28

BAB III ANALISA DAN DISAIN SISTEM 37

3.1. Analisa system 37

3.1.1. Analisa sistem ucapan 37

3.1.1.1. Normalisasi Text 37

3.1.1.2. Pembangkit Intonasi 39

3.1.2. Analisa kebutuhan hardware dan software 40

3.2. Metode perancangan 41

3.2.1. Perancangan form aplikasi 42

3.2.2. Perancangan diagram alir 43

3.2.3. Perancangan struktur basis data 44

3.2.3.1. Tabel inbox 45

3.2.3.2. Tabel phonebook 46

3.2.5. Perancangan antarmuka 46

BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM 49

4.1. Pendahuluan 49

4.2. Implementasi system 49

4.2.1. Kebutuhan hardware dan software 49

4.2.2. Tampilan Tukang Omong PD 50

4.2.3. Form utama 51

4.2.3.1. Tapsheet phonebook 52

4.2.3.2. Tapsheet inbox 52

4.2.3.3. Form about 54

4.2.3.4. Form petunjuk 54

4.3. Pengujian sistem 55

4.3.1. Pengujian koneksi antara ponsel dengan port USB pada komputer 55

4.3.2. Pengujian Pengiriman string pesan ke tukang Omong PD 56

BAB V PENUTUP 58

5.1. Kesimpulan 58

5.2. Saran 59

DAFTAR PUSTAKA 60

LAMPIRAN – LAMPIRAN

DAFTAR GAMBAR

BAB II DASAR TEORI

Gambar 2.1. Pengiriman SMS melewati SMSC 7

Gambar 2.2. Elemen dan arsitektur jaringan SMS 9

Gambar 2.3. Skenario proses MO-SM 11

Gambar 2.4. Skenario proses MT-SM 13

Gambar 2.5. Skema PDU untuk mengirim SMS ke SMSC 14

Gambar 2.6. Skema PDU untuk menerima SMS dari SMSC 20

Gambar 2.7. Hyper Terminal 26

Gambar 2.8. Subsistem Text To Speech 30

Gambar 2.9. Urutan proses konversi dari text ke ucapan 31

Gambar 2.10. Prinsip kerja generator sinyal ucapan Embrola 36

BAB III III ANALISA DAN DISAIN SISTEM

Gambar 3.1. Konfigufasi tetap aplikasi 39

Gambar 3.2. Tampilan setting konfigurasi dalam apliaksi 40

Gambar 3.3. Diagran alir menu utama 43

Gambar 3.5. Rancangan form utama 47

Gambar 3.6. Rancangan form about 47

Gambar 3.7. Rancangan form petunjuk 48

BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM

Gambar 4.1. Tampilan Tukang Omong PD 50

Gambar 4.2. Form utama 51

Gambar 4.3. Tapsheet phonebook 52

Gambar 4.4. Tapsheet inbox 53

Gambar 4.5. Memo isi SMS 53

Gambar 4.6. Form informasi 53

Gambar 4.7. Form about 54

Gambar 4.8. Form Petunjuk 54

DAFTAR TABEL

BAB II DASAR TEORI

Tabel 2.1. Daftar nomor SMSC 16

Tabel 2.2. Perhitungan waktu validitas SMS 17

Tabel 2.3. Skema 7 bit 19

Tabel 2.4. Skema 7 bit untuk teks'Pesan singkat' 19

Tabel 2.5. Skema Perubahan PDU ke Text untuk nilai D0F23CEC06 22

Tabel 2.6. Perintah AT untuk mengatur SMS 25

Tabel 2.7. Simbol-simbol flowchart diagram 27

BAB III III ANALISA DAN DISAIN SISTEM

Tabel 3.1. Tabel symbol dan karakter khusus 38

Tabel 3.2. Tabel inbox 45

Tabel 3.3. Tabel phonebook 46

BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM

Tabel 4.1. Hasil uji komunikasi antara komputer dengan ponsel 56

Tabel 4.2. Hasil pengujian pengiriman string pesan ke Tukang Omong PD..... 57

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Dengan berkembangan teknologi sekarang khususnya dalam telekomunikasi dengan adanya ponsel (telepon seluler), manusia dapat melakukan komunikasi tanpa terbatasi oleh ruang dan waktu sehingga mengefisiensikan suatu pekerjaan. Ponsel merupakan salah satu teknologi telekomunikasi tanpa kabel (wireless) yang mudah untuk dibawa kemana saja asalkan mendapatkan sinyal atau masih terjangkau oleh jaringan operator yang digunakan.

Dengan kemajuan teknologi sekarang ini bahkan ponsel sudah mendukung fitur untuk aplikasi Mp3 ,Radio, Video, Game dan WAP (Wireless Aplication Protocol), yang dapat dibeli dengan harga yang terjangkau sehingga menarik minat berbagai kalangan masyarakat untuk menggunakannya, hal ini terlihat dari jumlah pertumbuhan pengguna ponsel yang terus meningkat dari waktu ke waktu.

SMS adalah bagian dari salah satu fasilitas ponsel yang menyedia layanan komunikasi bagi pengguna untuk mengirim dan menerima informasi dalam bentuk pesan singkat. Layanan ini sangat efektif untuk memberitahukan informasi dalam waktu yang singkat, tetapi pesan SMS yang berupa teks membutuhkan waktu untuk membacanya. Untuk itu perlu dikembangkan pemikiran untuk mengubah teks menjadi ucapan, sehingga manusia tidak perlu membaca lagi namun cukup mendengarkan saja. Pengubahan teks menjadi ucapan bermanfaat bagi para penyandang *tuna netra* dan *tuna aksara* agar lebih mudah mendapat informasi dari suatu teks SMS.

Pengubahan teks menjadi ucapan dengan pembangkitan ucapan dilakukan menggunakan berbagai algoritma pengolahan sinyal digital yang dijalankan dengan perangkat lunak. Berkembangnya teknologi digital menyebabkan perkembangan sistem tulis-ucap atau TTS (Text to Speech) dan melahirkan beberapa alternatif baru untuk menjalankan bagian pembangkit ucapan untuk mencari pendekatan yang menghasilkan ucapan yang lebih alami.

TTS adalah suatu sistem perangkat keras maupun perangkat lunak yang mempunyai kemampuan untuk mengubah tulisan menjadi ucapan. Salah satu teknik yang digunakan pada TTS adalah teknik penyambungan diphone. Teknik ini dapat menghasilkan bunyi ucapan dengan tingkat kealamian yang tinggi. Penggunaan sistem TTS diharapkan dapat memberikan kemudahan dalam membaca SMS pada ponsel dengan menggunakan media komputer (http://www.indotts.com, September 2004).

1.2 RUMUSAN MASALAH

Bagaimana membaca SMS yang ada pada ponsel GSM, menyimpan dan merubah SMS tadi menjadi ucapan dengan menggunakan komputer.

1.3 BATASAN MASALAH

Dalam laporan akhir " Pengembangan Aplikasi Text To Speech Berbahasa Indonesia Sebagai Pembaca SMS Menggunakan Borland Delphi 7 " , penulis akan memberikan batasan-batasan masalah agar tidak terjadi penyimpangan maksud dan tujuan utama penyusunan skripsi ini. dengan batasan-batasan sebagai berikut:

1. Isi pesan SMS yang diambil pada ponsel berupa pesan teks yang tidak memuat gambar ataupun ringtone.

2. Menggunakan MBROLA Speech Engine (Tukang Omong PD V.1.03.0020) sebagai pembangkit ucapan dengan database suara (diphone) bahasa Indonesia. (www.kioss.com).
3. Tidak membahas komunikasi data GSM (Global System Mobile) secara terperinci.
4. Membaca Sms yang baru di inbox.
5. Pembuatan program dilakukan dengan menggunakan bantuan bahasa pemrograman Borland Delphi 7 dengan bantuan komponen XComm untuk antarmuka ponsel dengan PC.
6. Database Microsoft Office Access 2007.
7. Operating system windows.

1.4 TUJUAN PENELITIAN

Tujuan dari dibuatnya skripsi ini adalah membuat aplikasi yang dapat mengubah teks pesan SMS pada ponsel menjadi ucapan. Dengan menggunakan metode pengambilan pesan SMS pada ponsel melalui komputer, yang selanjutnya dengan menggunakan Tukang Omong PD V.01.03.0020 mengubah teks pesan SMS menjadi ucapan. Sistem ini diharapkan dapat memberikan kemudahan bagi seorang penyandang *tuna netra* dan *tuna aksara*, untuk mengetahui informasi pesan teks SMS pada ponsel dengan suara.

1.5 METODE PENELITIAN

Metode yang digunakan dalam penyusunan skripsi ini adalah :

1. Studi literatur, yaitu tinjauan pustaka untuk mempelajari teori-teori yang terkait dan berhubungan dengan permasalahan melalui media Referensi Buku, dan Literatur dari Internet.
2. Analisa Kebutuhan Sistem

   Data dan informasi yang telah diperoleh akan dianalisa agar didapatkan kerangka global yang bertujuan untuk mendefinisikan kebutuhan sitem baik hardware maupun software, di mana nantinya akan digunakan sebagai acuan perancangan sistem
3. Perancangan.

   Merancang dan mengembangkan kerangka global yang menggambarkan mekanisme dari sistem yang akan dibuat.

4 Coding.

Tahapan ini menerjemahkan hasil perancangan spesifikasi program dari tahapan sebelumnya kedalam baris–baris kode program yang dapat dimengerti oleh komputer.

5 Implementasi & Pengujian.

Pada tahap ini, sistem yang telah selesai dibuat akan diuji coba, yaitu pengujian berdasarkan fungsionalitas program, dan akan dilakukan koreksi dan penyempurnaan program jika diperlukan.

1.6 SISTEMATIKA PENULISAN

Untuk mempermudah dan memahami pembahasan penulisan skripsi ini, maka sistematika penulisan disusun sebagai berikut :

Bab I : PENDAHULUAN.

Berisi Latar Belakang, Rumusan Masalah, Batasan Masalah, Tujuan Penelitian, Metode Penelitian dan Sistematika Penulisan.

Bab II : LANDASAN TEORI.

Berisi tentang landasan teori mengenai permasalahan yang berhubungan dengan penelitian yang dilakukan.

Bab III : ANALISA DAN DISAIN SISTEM.

Dalam bab ini berisi mengenai analisa kebutuhan sistem baik software maupun hardware yang diperlukan untuk membuat kerangka global yang menggambarkan mekanisme dari sistem yang akan dibuat.

Bab IV : IMPLEMENTASI DAN PENGUAJIAN SISTEM.

Berisi tentang implementasi dari perancangan sistem yang telah dibuat serta pengujian terhadap sistem tersebut.

Bab V : PENUTUP.

Merupakan bab terakhir yang memuat intisari dari hasil pembahasan yang berisikan kesimpulan dan saran yang dapat digunakan sebagai pertimbangan untuk pengembangan penulisan selanjutnya.

BAB II

LANDASAN TEORI

2.1 SMS (*Short Message Service*)

SMS (Short Message Service) atau layanan pesan singkat mempunyai sejarah tersendiri sebagai media layanan yang paling meledak abad ini. Awalnya SMS berfungsi untuk memberikan layanan pengiriman pesan teks singkat antar handphone (telepon genggam/ telepon bergerak).

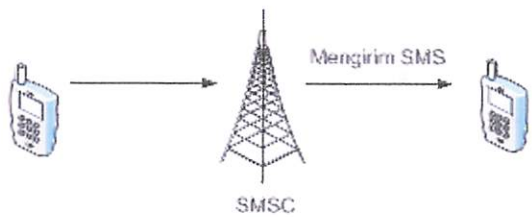
SMS sebetulnya hanya layanan tambahan terhadap layanan utama (layanan voice) dalam jaringan komunikasi GSM (Global System for Mobile Communication) mula-mula diperkenalkan tahun 1990. SMS dikembangkan dan distandarisasikan oleh ETSI (European Telecommunication and Standard Institute) sebuah lembaga telekomunikasi Eropa pada tahun 1991. ETSI mendefinisikan 2 standar SMS yaitu :

1. Layanan yang diperuntukkan antara dua pihak dengan memerlukan penghubung perantara dari titik ke titik atau pengirim dan penerima antar ponsel dinamakan SMSPP (SMS Point to Point).
2. Layanan broadcast antar network melalui satu atau lebih base station dengan seluruh pengguna berada di dalam sel atau daerah layanan dinamakan point to omni point atau SMSCB (SMS Cell Broadcast).

SMS dikirim dari ponsel pengirim ke ponsel penerima melewati SMSC (*Short Message Service Centre*). SMSC adalah perangkat lunak yang berada di jaringan operator telepon seluler dan mengatur proses yang menyangkut pengiriman pesan SMS dengan prinsip *store* dan *forward*. Pesan SMS yang dikirim ke SMSC akan disimpan terlebih dahulu hingga masa validitas tertentu terpenuhi jika pada saat SMS dikirim ponsel yang dituju sedang tidak aktif ataupun di luar jangkauan operator.

Setelah nomor ponsel yang dituju sudah terdeteksi aktif atau berada dalam jangkauan operator maka SMS akan diteruskan oleh SMSC kepada penerima (jika *expired period* belum terlampaui). Apabila SMS yang tersimpan di SMSC sudah melewati masa validitas yang ditentukan, SMS tersebut akan dihapus dan tidak akan diteruskan ke nomor ponsel yang dituju.

Dengan adanya SMSC, pengirim juga dapat mengetahui status dari SMS yang dikirim, apakah SMS tersebut sudah diterima atau belum. Gambar 2.1 menunjukkan proses pengiriman SMS melewati SMSC.



Gambar 2.1 Pengiriman SMS melewati SMSC [1]

Pada pengiriman dan penerimaan SMS ada 2 mode format SMS yang digunakan oleh operator.

1. Mode pertama adalah mode PDU (Protocol Data Unit) yaitu protokol data dalam suatu SMS, berupa pasangan-pasangan karakter ASCII (American Standard Code for Information Interchange) yang mencerminkan representasi angka heksadesimal dari informasi yang ada dalam suatu SMS, dengan panjang mencapai 160 (7 bit) atau 140 (8 bit) karakter yang mencerminkan bahasa I/O (kode). paket data pesan SMS dikemas bersama informasi tanggal, nomor tujuan, nomor pengirim, nomor operator, jenis skema SMS, masa valid SMS dan beberapa informasi lainnya (tergantung jenis paketnya).
2. Mode yang kedua adalah mode teks yang menggunakan format SMS dalam bentuk teks asli. Tidak semua operator GSM di Indonesia mendukung

format SMS mode teks dan kebanyakan menggunakan format SMS mode PDU.

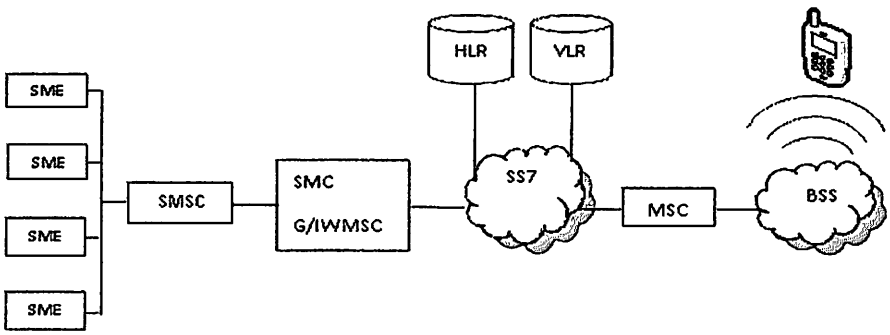
Dalam proses pengiriman pesan, dikenal dua jenis mobile yaitu ponsel pengirim (Mobile Originated) dan ponsel penerima (Mobile Terminated). Pada ponsel pengirim, metode yang digunakan adalah encodec. Metode ini mengubah SMS dalam bentuk teks menjadi format PDU yang kemudian dikirimkan ke SMSC. Pada ponsel penerima, metode yang digunakan adalah decodec yang merupakan kebalikan dari proses encodec. Pada proses decodec, format PDU yang diterima dari SMSC akan diubah menjadi format teks.

Pengiriman SMS dari dan ke PC perlu dilakukan terlebih dahulu koneksi ke SMSC. Koneksi PC ke SMSC dengan menggunakan terminal berupa GSM modem atau pun ponsel yang terhubung dengan PC. Dengan menggunakan ponsel Sony Ericsson T290i, SMS yang berasal dari atau ke SMSC harus berbentuk PDU.

PDU sendiri terdiri atas beberapa bagian yang berbeda antara mengirim dan menerima SMS dari SMSC yang telah diatur dan distandarisasi oleh ETSI. Format data PDU ini dikirimkan ke PC dalam bentuk teks (string) yang menunjukkan nilai heksadesimalnya. Jadi saat ponsel mengirim data heksadesimal F (0F h), maka yang diterima oleh PC adalah teks F.

2.1.1. Struktur Jaringan SMS

SMSC memiliki interkonektivitas dengan SME (Short Messaging Entity) yang dapat berupa jaringan e-mail, web, dan voice e-mail. SMSC inilah yang akan mengatur manajemen pesan SMS, baik untuk pengiriman, pengaturan antrian SMS, dan penerimaan SMS. Gambar 2.2 menunjukkan elemen dan arsitektur jaringan SMS



Gambar 2.2 Elemen dan arsitektur jaringan SMS [2]

Terdapat tujuh elemen pada jaringan SMS yaitu :

1. ***SME (Short Messaging Entity)***

   Adalah suatu piranti yang dapat menerima atau mengirim pesan singkat. SME berada dalam jaringan fixed, piranti bergerak, dan pusat layanan lainnya seperti VMS, web, dan e-mail.

2. ***SMSC (Short Message Service Centre)***

   SMSC merupakan perpaduan antara perangkat keras dan perangkat lunak yang bertugas untuk menerima, melakukan penyimpanan, dan meneruskan pesan pendek pada SME dan piranti bergerak.

3. ***SMS-GMSC (SMS-Gateway Mobile Switching Centre) dan SMS-IWMSC (SMS-Interworking Mobile Switching Centre)***

   SMS-GMSC merupakan suatu aplikasi yang dapat menerima pesan singkat dari SMSC, mengintrogasi HLR untuk menginformasikan routing, dan mengirimkan SMS ke MSC dari ponsel yang akan dituju. Aplikasi MSC yang dapat menerima pesan pendek dari jaringan bergerak dan mengirimkannya ke SMSC yang tepat merupakan tugas dari SMS-IWMSC.

4. ***HLR (Home Location Register)***

HLR pada elemen jaringan SMS merupakan suatu basisdata yang berfungsi sebagai penyimpanan tetap, pengelolaan data pelanggan, dan profil dari layanan. Setiap pelanggan bergantung pada sentral lokal pada jaringan tetap, tetapi pelanggan bergerak bergantung pada jaringan secara keseluruhan.

5. ***MSC (Mobile Switching Centre)***

   MSC melakukan fungsi pensaklaran sistem dan mengendalikan panggilan ke dan dari ponsel. MSC akan mengirimkan pesan pendek ke pelanggan tertentu melalui base station yang sesuai. MSC merupakan otak dari sistem radio selular. MSC juga berfungsi sebagai antarmuka antara jaringan GSM dengan public voice dan jaringan data.

6. ***VLR (Visitor Location Register)***

   VLR merupakan perangkat tempat penyimpanan data sementara (database temporer) dari pelanggan yang datang (visitor).

7. **BSS (Base Station System)**

   Semua fungsi yang berkaitan dengan transmisi sinyal elektromagnetis radio antara MSC dan perangkat bergerak dilakukan di BSS. Perangkat bergerak merupakan terminal wireless yang mampu menerima dan mengirimkan pesan pendek. BSS terdiri dari BSC (Base Station Controller) dan BTS (Base Transceiver Station).

   - BSC merupakan penyaklaran dengan kapasitas tinggi yang digunakan untuk melakukan fungsi-fungsi yang ada kaitannya dengan handover, manajemen jaringan radio, dan data konfigurasi dari sel. BSC mengontrol daya pancar radio baik pada base station ataupun ponsel.

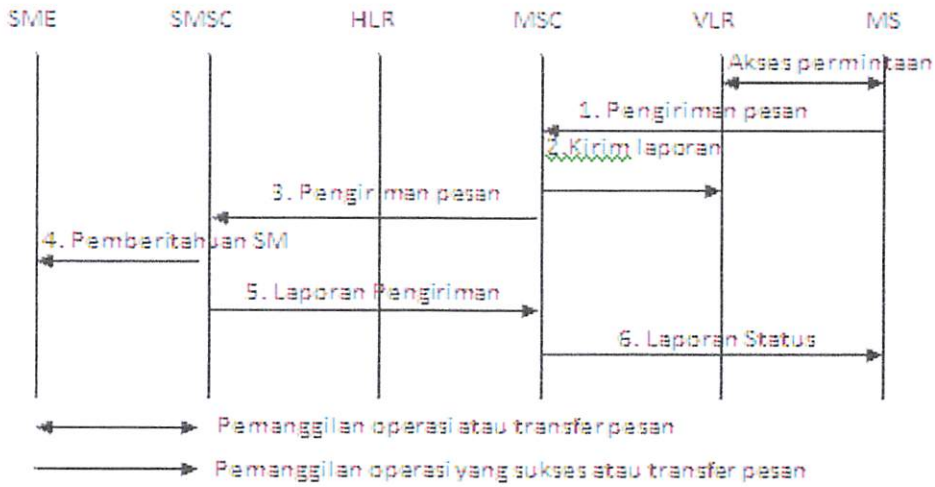
- BTS merupakan perangkat yang diperlukan oleh radio untuk mencakup satu atau beberapa sel serta menyediakan alur radio kirim dan terima.

2.1.2. Pengiriman dan Penerimaan SMS

Pada layanan SMSPP terhadap pelanggan, layanan SMS dibagi menjadi dua bagian yaitu:

1. ***MO-SM (Mobile Originated Short Message)***

   MO-SM dikirimkan dari handset yang MO-nya mampu menuju ke SMSC dan diterima ke pelanggan yang bergerak lainnya. Dalam layanan MO-SM selalu ada laporan yang dikirimkan ke handset, baik itu berupa konfirmasi pengiriman pesan pendek ke SMSC maupun konfirmasi kegagalan dalam pengiriman pesan dan pengidentifikasian penyebabnya.



Gambar 2.3 Skenario proses MO-SM [3]

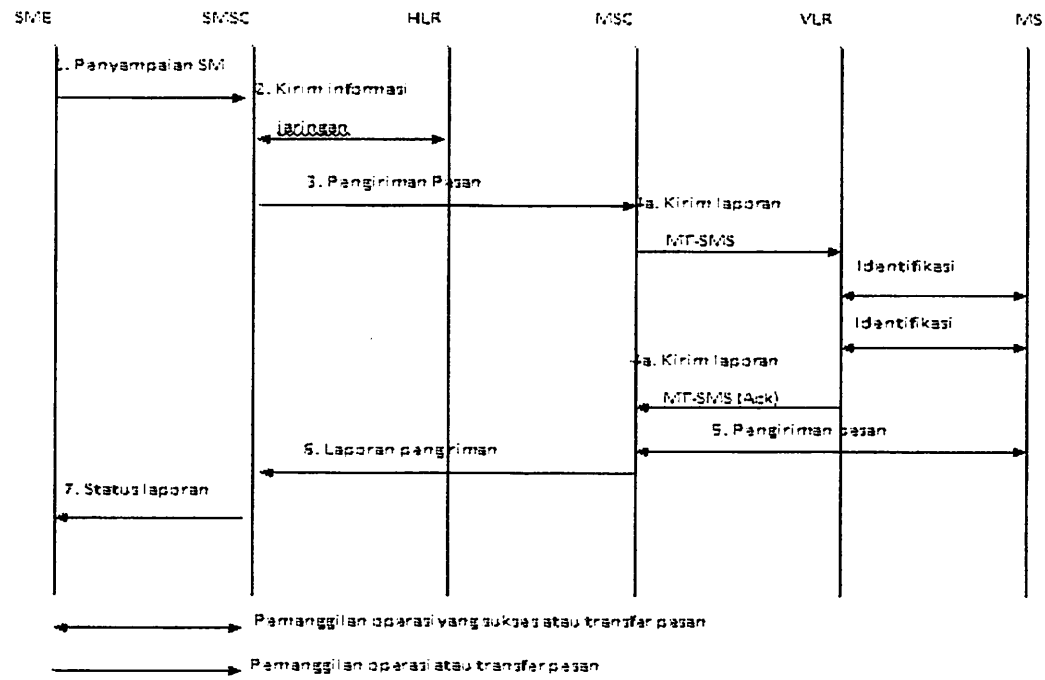
Penjelasan dari proses MO-SM adalah sebagai berikut:

a. MS akan mengirimkan pesan pendek ke MSC. Ketika proses ini terjadi, VLR akan melakukan akses permintaan informasi dari MS.

b. MSC akan mengecek VLR bahwa pengiriman pesan tersebut tidak melanggar permintaan layanan tambahan atau layanan mengenai batasan yang telah ditentukan berupa laporan.
c. MSC mengirim pesan pendek ke SMSC dengan menggunakan operasi kirim pesan pendek (Forward Short Message).
d. SMSC akan menyampaikan pesan pendek tersebut ke SME.
e. SMSC memberitahukan ke MSC dengan menggunakan laporan mengenai kesuksesan operasi kirim pesan pendek.
f. MSC akan mengembalikan hasil dari operasi MO-SM ke MS.

2. ***MT- SM (Mobile-Terminated Short Message)***

Kemampuan jaringan GSM mengirimkan SMS ke ponsel. Untuk laporan selalu dikirim kembali pada SMSC, salah satunya menegaskan konfirmasi penyampaian pesan pendek pada SMSC atau pemberitahuan pada SMSC apabila penyampaian pesan pendek mengalami kegagalan dan mengidentifikasi penyebab kegagalan tersebut. MT-SM dikirimkan dari SMSC ke handset dan dapat sampai ke pelanggan bergerak yang lain melalui MO-SM atau sumber lain seperti voice-mail. Gambar 2.4 menunjukkan proses MT-SM.



Gambar 2.4 Skenario proses MT-SM [3]

Penjelasan proses MT-SM sebagai berikut:

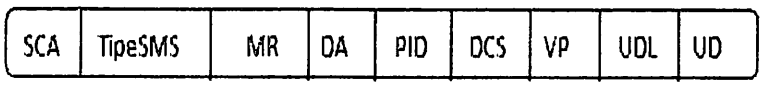
a. SME akan menyampaikan pesan singkat ke SMSC.
b. SMSC akan memeriksa HLR dan menerima informasi tentang rute (jalur) bagi pelanggan yang bergerak setelah menyelesaikan pengolahan proses internal.
c. SMSC mengirimkan pesan singkat ke MSC dengan menggunakan operasi pesan singkat.
d. MSC mengambil kembali informasi pelanggan dari VLR. Proses ini dapat melibatkan prosedur authentication atau identifikasi. Bila telah teridentifikasi, VLR akan mengirimkan balik laporan Acknowledgement-nya ke MSC.
e. MSC mengirimkan pesan singkat pada MS.

f. MSC akan mengembalikan hasil dari operasi kirim pesan singkat ke SMSC.

g. Jika SME meminta laporan status pengiriman, SMSC akan mengembalikan laporan status yang telah mengindikasikan pengiriman pesan singkat.

2.1.2.1. PDU untuk mengirim SMS ke SMSC

PDU untuk SMS yang dikirim ke SMSC terdiri atas delapan bagian. Gambar 2.5 menunjukkan skema PDU untuk mengirim SMS ke SMSC.



Gambar 2.5 Skema PDU untuk mengirim SMS ke SMSC [5]

Bagian-bagian tersebut adalah seperti berikut :

1. **SCA (*Service Center Address*)**

   Bagian ini merupakan bagian yang pertama dari PDU yang merupakan informasi dari alamat (nomor) SMSC. SCA ini tersusun atas tiga bagian yaitu *len, type of number*, dan nomor SMSC (*Service Center Number*).

   - *Len (Length)*

     Jumlah pasangan heksadesimal SMSC dalam bilangan heksadesimal.

   - Format nomor dari SMSC yang berisi kode nasional atau internasional SMSC (*type of number*)

     ✓ Kode nasional adalah 81

     Kode ini bukan merupakan kode yang biasa dipakai di Indonesia. Ini digunakan jika pada ponsel pengirim dipasang menggunakan kode nasional.

✓ Kode internasional adalah 91

Kode internasional ini di Indonesia merupakan kode yang biasanya digunakan (*default setting*).

o Nomor SMSC

Nomor SMSC dari operator pengirim pesan ini disusun sebagai pasangan heksadesimal dibolak-balik (heksadesimal disini sudah merupakan nomor SMSC itu sendiri yang sudah dalam bentuk *string*). Jika pada saat memasang-masangkan tersisa satu angka heksadesimal yang tidak memiliki pasangan, angka tersebut akan dipasangkan dengan heksadesimal F didepannya.

Di Indonesia pada umumnya bagian SMSC menggunakan kode internasional, sebagai contoh untuk nomor SMSC dari Indosat-M3 akan dikodekan dengan kode internasional dengan langkah pengkodean adalah seperti berikut :

a. Nomor SMSC dari Indosat-M3 adalah +62855000000. Jika dipasangkan maka masih tersisa satu angka yaitu angka 0 yang tidak memiliki pasangan, angka 0 tersebut akan dipasangkan dengan heksadesimal didepannya, sehingga jika dibalik akan menjadi 26 58 05 00 00 F0.
b. Kode internasional adalah 91.
c. Penggabungan kode internasional dan nomor SMSC yang dibalik menghasilkan pasangan 91 26 58 05 00 00 F0 yang berjumlah 7 pasang (07 h).
d. Jadi bagian nomor SMSC berisi 07912658050000F0.

Beberapa nomor SMSC operator seluler di Indonesia ditunjukkan oleh Tabel 2.1.

Tabel 2.1 Daftar nomor SMSC dalam format PDU dengan kode internasional.

No	Operator Seluler	SMSC	Kode PDU
1	Telkomsel (Simpati)	6281100000	06912618010000
2	Satelindo (Mentari)	62816124	059126181624
3	Excelcom (PRO-XL)	62818445009	0791261848540OF9
4	Indosat-M3	62855000000	07912658050000F0

2. Tipe SMS

Pada saat mengirim SMS, tipe SMS bernilai 1. Jadi bilangan heksadesimalnya adalah 01 untuk tipe SMS kirim.

3. Nomor referensi SMS (*Message Reference*)

Nomor referensi SMS merupakan acuan dari pengaturan pesan SMS. Nomor referensi SMS dibiarkan kosong terlebih dahulu karena nanti akan diberikan sebuah nomor referensi otomatis oleh ponsel. Maka bilangan heksadesimalnya adalah 00.

4. Nomor ponsel penerima (*Destination Address*)

Bagian ini berisi alamat nomor tujuan. Bagian nomor ponsel penerima juga tersusun atas tiga bagian yaitu panjang nomor tujuan (*Len*), format nomor tujuan (*Type Number*), dan nomor tujuan (*Destination Number*).

5. PID (*Protocol Identifier*) atau bentuk SMS

Bagian ini berisi tipe dari cara pengiriman dimana biasanya diatur dari ponsel pengirim, misalnya tipe teks standar, *Fax, E-mail, Telex,X400* dan lain–lain.

0 → 00 dikirim sebagai teks SMS

1 → 01 dikirim sebagai *telex*

2 → 02 dikirim sebagai *fax*

Untuk mengirim dalam bentuk teks akan digunakan kode heksadesimal 00.

6. DCS (*Data Coding Scheme*)

Bagian ini berisi skema *encoding* Data I/O.

Ada dua macam skema *encoding* yang digunakan yaitu :

a. Skema 7 bit → ditandai dengan angka 00.

b. Skema 8 bit → ditandai dengan angka yang lebih besar dari 0 lalu diubah ke heksadesimal.

Sebagian besar ponsel atau SMS *gateway* yang umum ada di pasaran menggunakan skema *encoding* 7 bit sehingga kode yang digunakan adalah 00.

7. VP (*Validity Period*)

Bagian ini berisi jangka waktu penyimpanan SMS di SMSC sebelum SMS *expired.* Jika bagian ini dikosongkan berarti tidak membatasi waktu berlakunya SMS sedangkan jika diisi dengan suatu bilangan integer yang kemudian diubah ke pasangan heksadesimal tertentu, bilangan tersebut akan mewakili jumlah waktu validitas SMS tersebut.

Tabel 2.2 Perhitungan waktu validitas SMS.

Nilai VP	Nilai Validitas Periode
0 - 143	(VP+1)*5 menit (interval 5 menit hingga 12 jam)
144 - 167	12 jam + ((TP-VP)*30 menit)
168 - 196	(VP - 166) * 1 hari
197 - 255	(VP - 192) * 1 minggu

Sebaiknya waktu validitas dipasang maksimum, supaya SMS pasti terkirim ke ponsel penerima. Untuk jangka waktu validitas SMS adalah 5 hari maka nilai VP = 166 + 5 = 171 d = AB h.

8. **UDL (*User Data Length*)**

Bagian ini berisi panjang SMS yang akan dikirim dalam bentuk teks standar. Misalkan contoh SMS yang dikirim adalah "Pesan singkat" yang mempunyai 13 karakter sehingga nilai heksadesimalnya adalah 0D h.

9. **UD (*User Data*)**

Bagian ini berisi isi SMS yang akan dikirimkan dan tentunya dalam format heksadesimal dengan skema pengkodean 7 bit. Skema 7 bit ini menunjukkan tabel konversi dari format bilangan 7 bit menjadi karakter yang diwakilinya dan sebaliknya. Format bilangan 7 bit ini kemudian diubah menjadi kode 8 bit untuk dikirimkan ke nomor tujuan. Skema 7 bit tersebut diperlihatkan pada Tabel 2.3.

Tabel 2.3 Skema 7 bit [5]

				b7	0	0	0	0	1	1	1	1
				b6	0	0	1	1	0	0	1	1
				b5	0	1	0	1	0	1	0	1
b4	b3	b2	b1		0	1	2	3	4	5	6	7
0	0	0	0	0	@	Δ	SP	0	¡	P	¿	p
0	0	0	1	1	£	_	!	1	A	Q	a	q
0	0	1	0	2	$	Φ	"	2	B	R	b	r
0	0	1	1	3	¥	Γ	#	3	C	S	c	s
0	1	0	0	4	è	Λ	¤	4	D	T	d	t
0	1	0	1	5	é	Ω	%	5	E	U	e	u
0	1	1	0	6	ù	Π	&	6	F	V	f	v
0	1	1	1	7	ì	Ψ	'	7	G	W	g	w
1	0	0	0	8	ò	Σ	(	8	H	X	h	x
1	0	0	1	9	Ç	Θ	)	9	I	Y	i	y
1	0	1	0	10	LF	Ξ	*	:	J	Z	j	z
1	0	1	1	11	Ø	ŋ	+	;	K	Ä	k	ä
1	1	0	0	12	ø	Æ	,	<	L	Ö	l	ö
1	1	0	1	13	CR	æ	-	=	M	Ñ	m	ñ
1	1	1	0	14	Å	β	.	>	N	Ü	n	ü
1	1	1	1	15	å	É	/	?	O	§	o	à

Berikut contoh pengkodean dengan skema pengkodean 7 bit untuk mengubah teks "Pesan singkat", kemudian menjadi kode 8 bit untuk dikirimkan yang ditunjukkan Tabel 2.4

Tabel 2.4 Skema pengkodean 7 bit untuk teks "Pesan singkat".

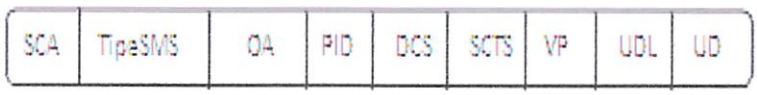
Nilai	Dec	*Septet* (7bit)	*Oktet* (8bit)	Hasil
P	80	101 0000	1101 0000	D0
e	101	110 0101	1111 0010	F2
s	115	111 0011	0011 1100	3C
a	97	110 0001	1110 1100	EC
n	110	110 1110	0000 0110	06
spasi	32	010 0000	1100 1101	CD
s	115	111 0011	1101 0011	D3
i	105	110 1001		
n	110	110 1110	1110 1110	EE
g	103	110 0111	1111 0011	F3
k	107	110 1011	0011 1010	3A
a	97	110 0001	0100 1100	4C
t	116	111 0100	0000 0111	07

Data skema 7 bit dari teks SMS yang dikirim, dideret sedemikian rupa, kemudian mengambilnya tiap 8 bit. Jika pada akhir pengambilan bit yang tersisa kurang dari 8 bit, maka di bagian depan ditambah dengan nol hingga berjumlah 8 bit. Jadi PDU untuk teks "Pesan singkat" adalah D0F23CEC06CDD3EEF33A4C07 ditambah dengan panjang teks 13 (0D h). Jadi PDU isi SMS-nya adalah 0DD0F23CEC06CDD3EEF33A4C07.

Jadi pengiriman teks "Pesan singkat" ke nomor 628540787149 lewat SMSC Indosat-M3 tanpa ada jangka waktu expired, maka PDU lengkapnya adalah sebagai berikut 07912658050000F001000C912658048717940000 0DD0 F23CEC06CDD3EEF33A4C07 (Candra,Jakarta 2004).

2.1.2.2. PDU SMS yang diterima dari SMSC

PDU untuk SMS yang diterima dari SMSC terdiri atas delapan bagian yang sebagian besar hampir sama dengan PDU untuk SMS yang dikirim. Gambar 2.6 menunjukkan skema PDU untuk menerima SMS dari SMSC.



Gambar 2.6 Skema PDU untuk menerima SMS dari SMSC [4]

Berikut adalah penjelasan dari kedelapan bagian tersebut:

1. **SCA**

Sama dengan untuk PDU SMS yang dikirim. SCA memiliki 3 bagian yaitu jumlah pasangan heksadesimal SMSC dalam bilangan heksadesimal, format nomor dari SMSC yang berisi kode nasional atau internasional SMSC, dan nomor SMSC dari operator pengirim untuk SMSC. Operator seluler

Telkomsel dengan nomor SMSC 6281100000, maka PDU bagian SCA adalah 06912618010000.

2. **Tipe SMS**

Tipe SMS untuk SMS terima adalah 04 h.

3. **OA (*Originator Address*) atau nomor ponsel pengirim**

OA berisi alamat nomor pengirim. Bagian nomor ponsel pengirim juga tersusun atas tiga bagian seperti pada PDU untuk nomor ponsel penerima yaitu panjang nomor pengirim, format nomor pengirim, dan nomor pengirim (*Originator Number*). Bentuk nomor ponsel pengirim sama dengan nomor ponsel penerima untuk PDU SMS yang dikirim.

4. **PID atau bentuk SMS**

Bagian ini berisi tipe dari cara pengiriman dimana biasanya diatur dari ponsel pengirim, misalnya tipe teks standar, *fax*, *telex*, dan lain–lain seperti halnya pada SMS PDU pengirim. Biasanya bagian ini memiliki nilai 00 yang menunjukkan bahwa bentuk SMS teks.

5. **DCS atau skema *encoding***

Biasanya bagian ini memiliki nilai 00 yang menunjukkan bahwa skema pengkodean yang digunakan adalah skema 7 bit.

6. **SCTS (*Service Centre Time Stamp*)**

SCTS adalah tanggal dan waktu penerimaan SMS oleh SMSC penerima. Pada bagian ini memiliki 14 bilangan heksadesimal (7 pasangan), 6 pasangan yang berarti yy/mm/dd hh:mm:ss (tahun/ bulan/ tanggal jam: menit : detik) dan 1 pasangan yang menunjukkan kawasan waktu berdasarkan GMT,

sebagai contoh data PDU SMS yang diterima pada bagian tanggal dan waktu SMS sampai SMSC 40807101010082 = 04/08/17 10:10:00 28 = 17 Agustus 2004, pukul 10:10:00 WIB.

7. **Isi SMS**

Bagian ini terdiri atas dua bagian yaitu :

a. UDL

UDL merupakan panjang isi pesan yang diterima dalam bentuk teks standar (jumlah huruf dari isi). Misalnya untuk kata "Pesan", ada 5 huruf. Jadi pada bagian ini dituliskan 05.

b. UD

UD merupakan isi pesan yang diterima berrupa pasangan bilangan heksa. Data ini menyerupai bilangan ASCII dari bilangan alphanumerik, namun beberapa karakter oleh forum SMS sedunia diadakan perubahan.

Cara mengubah data SMS ke dalam karakter yang diwakilinya adalah sebagai berikut: contoh data isi SMS adalah 05D0F23CEC06. Kode 05 menunjukkan jumlah karakter yang dikirim, data selanjutnya adalah kode dari karakter yang dikirimkan. Tabel 2.5 menunjukkan skema pengubahan PDU ke teks untuk nilai D0F23CEC06.

Tabel 2.5 Skema pengubahan PDU ke teks untuk nilai D0F23CEC06.

Nilai	*Oktet (8bit)*	*Septet (7 bit)*	*Hasil*
D0	*1101 0000*	*101 0000*	*P*
F2	*1111 0010*	*110 0101*	*e*
3C	*0011 1100*	*111 0011*	*s*
EC	*1110 1100*	*110 0001*	*a*
06	*0000 0110*	*110 1110*	*n*

Dengan melihat data 7 bitnya dan melihat karakter alphanumeriknya pada Tabel 2.5 maka dapat diperoleh teks untuk PDU D0F23CEC06 adalah "Pesan". Jika diterima PDU SMS seperti 06912618010000 04 0C91261822056896000040505101010082005D0F23CEC06 maka dapat dianalisis seperti berikut :

- Kode 06912618010000 mempunyai arti bahwa pengirim menggunakan SMSC Telkomsel (Simpati).
- Kode 04 menunjukkan bahwa PDU ini adalah PDU SMS terima.
- Kode 0C91261822056896 menunjukkan bahwa nomor tujuan memiliki panjang 12 digit (0C) yaitu 628122508669, sedangkan 91 menunjukkan kode internasional sehingga perlu ditambah kode negara (Indonesia = 62).
- Kode 0000 menunjukkan bentuk SMS adalah teks dan skema pengkodeannya adalah 7 bit.
- Kode 40505101010082 menunjukkan SMS sampai di SMSC pada tanggal 15-05-04 pukul 10:10:00 dan 28 menunjukkan kawasan waktu Indonesia.
- Kode 00 menunjukkan bahwa SMS tidak memiliki batas waktu validitas.
- Kode 05D0F23CEC06 menunjukkan banyaknya karakter yang dikirim adalah 5 karakter yaitu kata "Pesan" yang dikodekan dengan D0F23CEC06.

2.2. Perintah AT (*AT Command*)

Perintah AT merupakan media komunikasi antara ponsel dengan PC. Perintah AT ini dapat digunakan untuk menulis, mengirim, membaca SMS, dan mengambil data yang ada pada ponsel maupun menjalankan aplikasi tertentu di ponsel. Antara ponsel dan komputer diperlukan kabel data untuk melakukan perintah AT. Perintah ini mengacu pada spesifikasi ETSI (European Telecommunication and Standard Institute) GSM 07.07 dan GSM 07.05 atau sesuai dengan spesifikasi yang diberikan oleh perusahaan pembuat ponsel (*http://www.etsi.org,juli 2006*) .

Perintah AT sebenarnya hampir sama dengan perintah > (*prompt*) pada DOS (*Disk Operating* System). Perintah-perintah yang dimasukkan ke *port* dimulai dengan kata AT, lalu diikuti oleh karakter lainnya yang mempunyai fungsi-fungsi unik. Beberapa contoh perintah AT yang penting untuk mengatur SMS dapat dilihat pada Tabel 2.6. Penjelasan secara lebih lengkap mengenai penjelasan fungsi-fungsi perintah AT ponsel Sony Ericsson T290i dan cara penggunaannya dapat dilihat di lampiran laporan Tugas Akhir ini, sehingga dapat dicoba perintah AT yang diperlukan untuk aplikasi yang diinginkan.

Sebenarnya terdapat dua cara dalam pengiriman dan penerimaan pesan SMS yaitu mode teks dan mode PDU tergantung dari fasilitas yang ada pada ponsel yang digunakan. ponsel Sony Ericsson T290i mendukung dua mode tapi kebanyakan ponsel di Indonesia hanya mendukung mode PDU, Pada mode PDU pesan dikirim dan diterima berisi PDU *string* yang tidak hanya berisi pesan saja tetapi juga beberapa bagian lain seperti SMSC. Keuntungan dari mode PDU yaitu dapat melakukan encoding sendiri yang harus didukung pula oleh perangkat keras dan operator GSM, selain itu dengan mode PDU dapat melakukan kompresi data, menambahkan nada

dering, dan gambar pada pesan yang akan dikirimkan. Tabel 2.6 menunjukkan beberapa perintah AT yang digunakan untuk mengatur SMS.

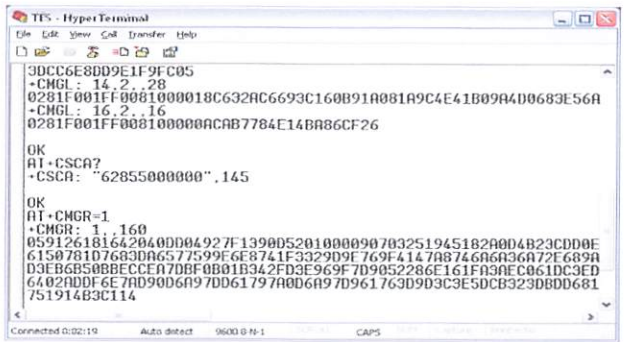
Tabel 2.6 Beberapa perintah AT untuk mengatur SMS.

Perintah AT	Keterangan
AT+CMGR=<mem>(Enter)	Digunakan untuk membaca SMS pada lokasi memori tertentu. Lokasi memori ini ditunjukkan oleh <mem>. Jadi AT+CMGR=1 menunjukkan sms yang dibaca adalah SMS pada lokasi memori 1
AT+CMGL=<stat>(Enter)	Digunakan untuk membaca SMS dengan status tertentu yang ditunjukkan oleh <stat> dan ditampilkan dalam bentuk daftar. <stat> : 0 = SMS terima belum terbaca 1 = SMS terima sudah terbaca 2 = SMS tersimpan belum dikirim 3 = SMS tersimpan dan sudah terkirim 4 = semua SMS
AT+CMGD=<mem>(Enter)	Digunakan untuk menghapus SMS pada lokasi memori tertentu yang ditunjukkan oleh <mem>
AT+CMGF=<mode>	Digunakan untuk mengatur format SMS yang ditunjukkan oleh <mode>. <mode> : 0 = PDU mode 1 = Teks mode
AT+CSCA=?	Digunakan untuk mengetahui nomor SMSC

Beberapa perintah AT yang terdapat pada Tabel 2.6 digunakan untuk menulis, mengirim, dan membaca SMS. Perintah AT+CMGR=1 digunakan untuk membaca SMS pada lokasi memori dengan indeks 1 pada inbox ponsel. Perintah

AT+CMGL=0 digunakan untuk membaca SMS dengan status belum terbaca dan ditampilkan dalam bentuk daftar.

Untuk dapat mencoba perintah AT bisa menggunakan perangkat lunak Hyper Terminal yang telah disediakan oleh OS (Operation System) Windows, Hyper Terminal terletak pada menu (All Programs – Accessories, Communications – HyperTerminal). Gambar 2.7 menunjukkan cara penggunaan perintah AT dengan menggunakan perangkat lunak Hyper Terminal.



Gambar 2.7 Hyper Terminal

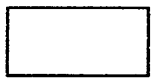
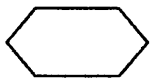
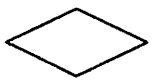
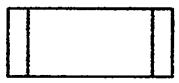
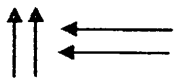
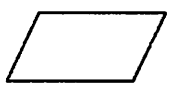
Hyper Terminal merupakan aplikasi yang dapat digunakan untuk mengirimkan perintah AT dari komputer ke ponsel dan menerima respon dari ponsel. Data pada ponsel dapat diketahui dengan mengirimkan perintah AT yang sesuai, sebagai contoh perintah AT+CMGR=1 digunakan melihat data PDU pada kotak masuk ponsel dengan indek 1. Hyper Terminal akan menerima respon "OK" dari ponsel apabila data pada ponsel dapat diambil.

2.3. Bagan Alir Program (Flowchart)

Suatu metode untuk mengambarkan tahap – tahap pemecahan masalah adalah dengan mempresentasikan simbol – simbol tertentu yang mudah di mengerti dan standart. Program flowchart adalah salah satu yang dapat didefinisikan sebagai bagan

yang menjelaskan secara rinci langkah – langkah dari proses program (Jogiyanto, HM : 2001 : 802).

Tabel 2.7 Simbol – Simbol Flowchart Diagram

Simbol	Nama simbol	Keterangan
	Terminal	Untuk menggambarkan awal dan akhir proses aliran dokumen
	Processing	Dipakai untuk pengolahan aritmatika dan pemindahan data
	Preparation	Dipakai untuk memberikan nilai awal dari suatu variable atau counter
	Decision	Dipakai untuk mewakili operasi perbandingan logika / keputusan
	Predefined process	Dipakai untuk proses yang detailnya djelaskan secara terpisah , misalnya dalam bentuk subroutine
	Garis alir	Dipakai untuk menunjukkan aliran dari program
	Penghubung	Menunjukkan penghubung di halaman yang masih sama atau dihalaman lainnya
	Input / Output	Mewakili data input atau output

2.4. TTS (Text To speech)

Text To Speech bahasa Indonesia ini diberi nama Tukang omong PD yang dibuat oleh Team Kioss Project (by Lury Darmawan), aplikasi ini sekarang sudah tidak dikembangkan lagi oleh KIOSS dan terakhir update Last Update : 2003-01-30 10:46 Version: 1.03.16, Aplikasi Tukang Omong menggunakan Speech Engine MBrola dan database suara (diphone) milik DR. Arry Akhmad Arman (Doctor dari ITB) yang ia buat tahun 2000 di Belgia. Aplikasi ini bisa membaca teks seperti "10000" dibaca "Sepuluh Ribu", selain itu juga beberapa expresi matematika, ditambah lagi dia bisa bicara tentang jam, misalnya kalau ditanya sekarang jam berapa maka dia akan menjawab. Dia juga akan ngomong sekarang jam berapa setiap setengah atau 1 jam tergantung settingan kita. Salah satu teknik yang digunakan pada TTS adalah teknik penyambungan diphone, teknik ini dapat menghasilkan bunyi ucapan dengan tingkat kealamian yang tinggi.

Pesan atau informasi yang dikirimkan lewat ucapan memiliki kelebihan antara lain :

1. Pengguna dapat dengan mudah memahami pesan atau informasi tanpa perlu intensitas konsentrasi tinggi.
2. Pesan atau informasi dapat diterima saat pengguna sedang terlibat dengan aktivitas lain, misalnya saat menangani atau sedang melihat objek lain.

Pengubahan Teks menjadi ucapan

Sistem Text to Speech pada prinsipnya terdiri dari dua subsistem dasar, yaitu:

1. Subsistem konverter teks ke fonem

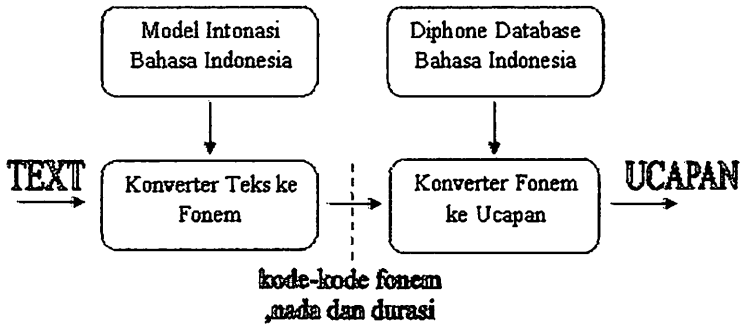
Subsistem konverter teks ke fonem yang memiliki dua fungsi utama. Pertama adalah mengambil kalimat masukan dalam suatu bahasa tertentu yang berbentuk barisan teks dan mengubah beberapa hal seperti nomor dan tanda ke dalam tulisan sesuai dengan bunyi yang seharusnya, atau sering disebut dengan normalisasi teks (text normalization). Kemudian menentukan kode fonetik (phonetic transcriptions) untuk tiap kata beserta durasi dan nadanya.

Kode fonem adalah kode yang merepresentasikan unit bunyi yang ingin diucapkan. Pengucapan kata atau kalimat pada prinsipnya adalah urutan bunyi atau secara simbolik adalah urutan kode fonem.

2. Subsistem konverter fonem ke ucapan

Subsistem konverter fonem ke ucapan yang akan menerima masukan kode-kode fonem serta nada dan durasi yang telah dihasilkan oleh bagian sebelumnya. Berdasarkan kode-kode tersebut bagian ini akan menghasilkan bunyi atau sinyal ucapan yang sesuai dengan kalimat yang ingin diucapkan. Ada beberapa alternatif teknik yang dapat digunakan untuk implementasi bagian ini. Salah satu teknik yang digunakan adalah penyambungan diphone.

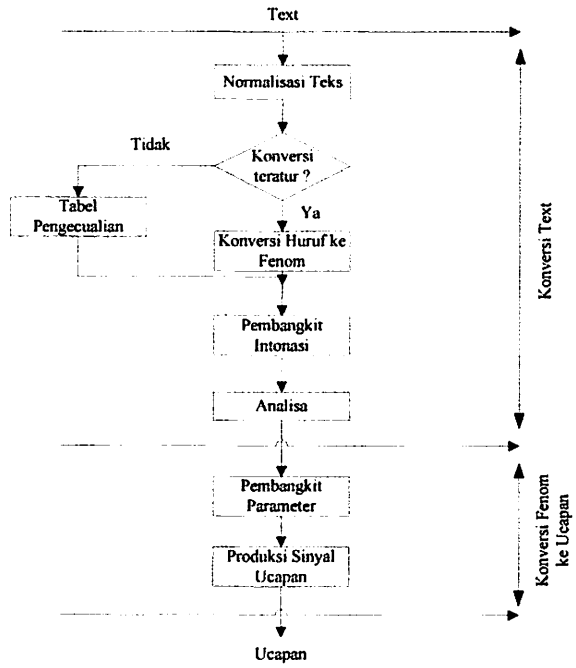
Pada sistem yang menggunakan teknik penyambungan diphone, sistem harus didukung oleh suatu basis data diphone yang berisi rekaman segmen-segmen ucapan yang berupa diphone. Ucapan dari suatu bahasa dibentuk dari satu set bunyi mungkin berbeda untuk setiap bahasa, oleh karena itu setiap bahasa harus dilengkapi dengan basis data diphone yang berbeda



Gambar 2.8. Subsistem text to speech [14]

Setiap fonem dilengkapi dengan informasi durasi dan nada. Informasi durasi diperlukan untuk menentukan berapa lama suatu fonem diucapkan, sedangkan informasi nada diperlukan untuk menentukan tinggi rendahnya nada pengucapan suatu fonem. Durasi dan nada bersama-sama akan membentuk intonasi suatu ucapan.

Setiap bahasa memiliki aturan cara pembacaan dan cara pengucapan teks yang sangat spesifik. Hal ini menyebabkan implementasi unit konverter teks ke fonem menjadi sangat spesifik terhadap suatu bahasa. Tahapan-tahapan utama konversi dari teks menjadi ucapan dapat dilihat pada Gambar 2.9.



Gambar 2.9 Urutan proses konversi dari teks ke ucapan [14].

Tahapan-tahapan tersebut ialah normalisasi teks, konversi huruf ke fonem dan pengecualian, pembangkitan intonasi, analisis fonetik, dan generator sinyal ucapan MBROLA. Berikut ini adalah penjelasan dari masing-masing tahapan.

Normalisasi Teks

Tahap pertama dari sistem konversi teks ke ucapan adalah tahap normalisasi teks. Pada tahap ini semua teks kalimat diubah menjadi teks yang secara lengkap memperlihatkan cara pengucapannya. Normalisasi teks meliputi pengubahan singkatan, akronim, angka, tanggal, waktu, karakter-karakter khusus, dan simbol-simbol dengan bentuk huruf alphabet lengkap sehingga tidak terjadi ambiguitas berkenaan dengan cara pengucapan (Dutoit T,Dordrecht,1997).

Berkaitan dengan pengubahan teks kalimat, tahapan normalisasi teks sendiri terdiri dari beberapa blok bagian meliputi :

- ***konverter angka***, bagian konverter angka berfungsi mengubah angka menjadi deretan huruf alphabet yang menggambarkan cara pengucapannya, sebagai contoh angka 14550 akan diubah menjadi empat belas ribu lima ratus lima puluh.
- Bagian ***konverter akronim***, berfungsi mengubah akronim menjadi huruf tunggal atau deretan huruf alphabet yang menggambarkan cara pengucapannya, sebagai contoh singkatan "dlm" akan diubah menjadi "dalam".
- Bagian ***konverter simbol*** dan ***karakter khusus*** akan mengubah karakter dan simbol menjadi format teks sesuai dengan cara pengucapannya, sebagai contoh simbol "%" pada kalimat "3%" akan terbaca "tiga persen".

Subsistem ini harus memiliki pustaka setiap unit ucapan dari suatu bahasa. Pengubahan angka, akronim, singkatan, simbol, dan karakter khusus sangat tergantung pada basisdata yang digunakan sistem. Ucapan dari suatu bahasa dibentuk dari satu set ucapan bunyi yang mungkin berbeda untuk setiap bahasa, oleh karena itu setiap bahasa harus dilengkapi dengan basisdata yang berbeda.

Konversi huruf ke fonem dan pengecualian

Tahap selanjutnya setelah normalisasi teks adalah proses konversi huruf ke fonem. Tahapan ini bertujuan untuk mendapatkan ketentuan pengucapan dasar dari setiap kata teks yang telah

dinormalisasi. Konversi huruf menjadi fonem biasanya dilakukan dengan dua pendekatan dasar yaitu :

1. **Pendekatan berdasarkan kamus.**

   Pendekatan ini menggunakan suatu pendekatan yang menggunakan sebuah kamus besar dari sebuah bahasa yang terdiri dari semua kata, dan pengucapannya yang benar kemudian disimpan pada sebuah program.

   Pendekatan ini memiliki keuntungan lebih cepat dan teliti dalam proses pengubahan huruf ke fonem tetapi akan berbeda jika kata yang diberikan tidak ada dalam kamus. Selain itu kamus memiliki perbendaharaan kata yang sangat banyak sehingga dibutuhkan ruang memori yang cukup besar untuk penggunaannya.

2. **Pendekatan menurut aturan (Rule-based approach).**

   Pendekatan ini menggunakan aturan yang telah ditentukan untuk pengucapan kata berdasarkan ejaannya. Pendekatan ini bekerja pada masukan apa saja, tetapi kerumitan dari aturan berkembang sehingga membutuhkan ejaan dan pengucapan yang besarnya tidak tentu.

   Pendekatan ini dapat diimplementasikan dengan tabel konversi yang berisi pasangan antara urutan huruf dan urutan fonem, bahkan mungkin hanya berisi satu huruf dan satu fonem. Aturan yang lebih sulit biasanya diimplementasikan dengan tabel konversi yang akan diterapkan jika kondisi rangkaian huruf tetangga kiri dan tetangga kanannya terpenuhi. Contoh

bentuk aturan konversi huruf ke fonem yang memenuhi teknik tersebut adalah sebagai berikut[14]:

teks-kiri[deretan huruf] teks-kanan = deretan fonem

Huruf tertentu yang ditunjuk dalam posisi [deretan huruf] akan diubah menjadi fonem dalam "deretan fonem" jika teks kiri dan teks kanan terpenuhi.

Bahasa Inggris termasuk bahasa yang mempunyai keteraturan yang rendah untuk proses konversi teks ke fonem. Suatu sistem TTS bahasa Inggris biasanya dilengkapi dengan suatu basis data yang berisi ribuan kata serta konversi padanan urutan fonemnya.

Bahasa Indonesia termasuk bahasa yang jelas aturan konversinya. Sebagian besar kata dalam bahasa Indonesia dapat diubah menjadi fonem dengan aturan yang jelas dan sederhana, walaupun tetap ada kondisi-kondisi yang tidak dapat ditemukan keteraturannya, sebagai contoh simbol huruf e dapat diucapkan sebagai e pepet atau e taling, artinya harus dikonversikan menjadi fonem yang berbeda untuk kondisi yang berbeda. Kondisi yang masih dapat ditangani oleh aturan diimplementasikan dengan blok konversi huruf ke fonem. Konversi yang tidak teratur ditangani oleh bagian tabel pengecualian.

Pembangkitan Intonasi

Gejala intonasi atau prosodi, mempunyai hubungan yang erat dengan struktur kalimat dan interelasi kalimat dalam suatu wacana.

Karena sifatnya yang sangat subyektif, pembahasan intonasi lebih ditekankan pada aspek gramatikal dibandingkan aspek emosional.

Sebuah ucapan dilafalkan oleh penuturnya dalam pola melodi tertentu yang diterima pendengarnya sebagai sebuah deret nada dengan tinggi yang berbeda. Tingkat tinggi nada tidak dapat ditentukan secara pasti, sangat bervariasi tergantung dari jenis kelamin, emosi atau sikap.

Prosodi adalah perubahan nilai-nilai nada atau frekuensi dasar selama pengucapan kalimat dilakukan atau nada sebagai satuan waktu. Pada prakteknya, informasi pembentuk prosodi berupa data nada serta durasi pengucapannya untuk setiap fonem yang dibangkitkan. Prosodi bersifat sangat spesifik untuk setiap bahasa, sehingga model yang diperlukan untuk membangkitkan data-data prosodi menjadi sangat spesifik untuk setiap bahasa. Beberapa metode pemodelan intonasi diantaranya adalah Tone Sequence Model, yang membangkitkan intonasi dengan cara merepresentasikan suatu kalimat dengan kuat lemahnya suku kata dan urutan tinggi rendahnya nada berkaitan dengan tekanan suku kata dan jeda prosodi.

Analisis Fonetik

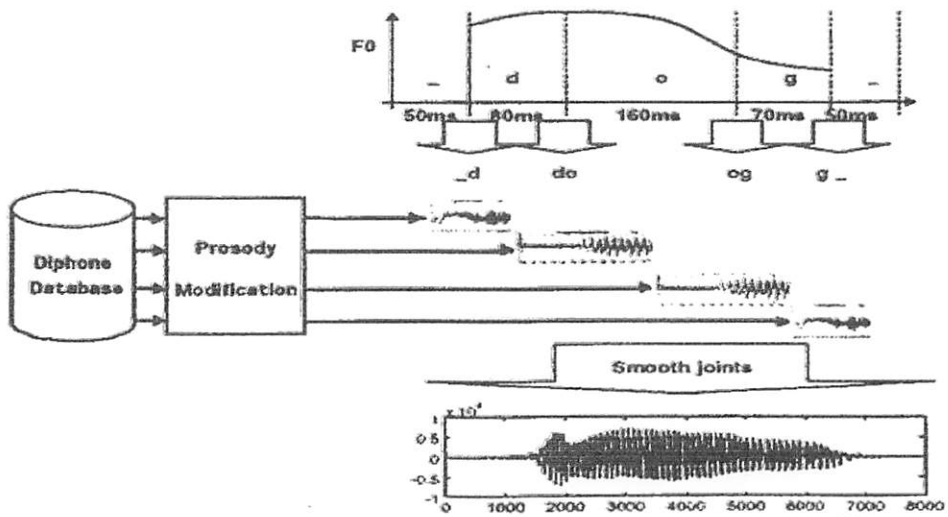
Satu tahap berikutnya yang masih sering dilakukan adalah analisis fonetik. Tahap ini dapat dikatakan sebagai tahap penyempurnaan, yaitu melakukan perbaikan di tingkat bunyi, sebagai contoh dalam bahasa Indonesia, fonem /k/ dalam kata "bapak" tidak

pernah diucapkan secara tegas, atau adanya sisipan fonem /y/ dalam pengucapan kata "alamiah" antara fonem /i/ dan /a/.

Generator sinyal ucapan MBROLA

Salah satu contoh sistem pembangkit ucapan yang cukup popular adalah MBROLA yang dikembangkan oleh Faculte Polytechnique de Mons TCTS Lab, Belgia. MBROLA pada dasarnya merupakan sistem yang berbasis pada penyambungan diphone. MBROLA menerima masukan parameter berupa fonem dan informasi intonasi dengan format khusus.

Basisdata diphone merupakan komponen yang penting dalam MBROLA. Basisdata ini diproses dari segmen ucapan yang nantinya digunakan dalam proses pembangkitan ucapan. Basisdata diphone MBROLA telah tersedia untuk bahasa Portugis, Inggris, Belanda, Jerman, Prancis, dan juga Indonesia.



Gambar 2.10 Prinsip kerja generator sinyal ucapan MBROLA [15].

BAB III

ANALISA DAN DISAIN SISTEM

3.1 ANALISIS SISTEM

3.1.1 Analisis sistem ucapan

Aplikasi pembaca sms ini menggunakan Speech Engine MBrola yang dibuat oleh Arry Arman dan telah dikembangkan oleh Lury Darmawan menjadi Tukang Omong "PD" V.1.03.0020. (Copyright by Lury Darmawan at WWW.Kioss.Com, September 2004).

3.1.1.1 Normalisasi Teks

Tukang Omong PD V.1.03.0020 mempunyai spesifikasi meliputi pengubahan singkatan, akronim, angka, tanggal, waktu, karakter-karakter khusus dan simbol-simbol dengan huruf alphabet sesuai dengan database yang dibuat oleh Lury Darmawan.

Konversi huruf mengunakan suatu pendekatan sebuah kamus besar dari bahasa indonesia yang tediri dari semua kata berdasarkan ejaannya yang baku dan benar. Sebagian besar kata dalam bahasa Indonesia dapat di ubah menjadi fonem dengan aturan yang jelas dan sederhana, walaupun tetap ada kondisi – kondisi yang tidak dapat ditemukan keteraturannya. Karena sifatnya yang sangat kompleks, pembahasan konversi huruf lebih ditekankan pada pendekatan kamus bahasa indonesia yang benar. Setiap suku kata teks isi pesan tidak bisa ditentukan secara pasti, sangat tergantung dari kebiasaan pengirim.

Berkaitan dengan pengubahan teks kalimat dengan keterbatasan kemampuan aplikasi dan Tukang omong "PD" v.1.03.0020 setelah dilakukan pengujian penggunaan aplikasi, tahapan normalisasi teks terdiri dari beberapa blok bagian meliputi :

- **Konverter angka**

  Aplikasi ini dengan bantuan Tukang Omong PD V.1.03.0020 mampu mengubah angka 0 – 999999999999999 (nilai nol sampai nilai tilyurn) dengan batasan 15 angka.

- **Konverter huruf**

  Aplikasi ini dengan bantuan tukang omong PD V.1.03.0020, mampu mengubah semua huruf vokal dari "a i u e o" dan semua huruf konsonan a – z, kecuali huruf " x q" yang diimplentasikan hanya satu huruf. Disamping itu aplikasi ini mempunyai kelemahan dalam konverter huruf vokal/konsonan yang diulang. Sebagai contoh pengulangan huruf "aa" atau "bb", maka aplikasi ini akan lagsung memberi peringatan error karena keterbatasan database yang dibuat.

- **Konverter simbol dan karakter kusus**

Tabel 3.1 Tabel simbol dan karakter khusus

Simbol	Keterangan	Simbol	Keterangan
~	Tak terhingga	)	Kurung tutup
`	Pel 'i	-	Kurang
@	At	_	Garis bawah
#	Kres	+	Tambah
$	Dolar	=	Sama dengan
%	Persen	:	Titk dua
&	Dan	'	Petik
*	Bintang	<	Lebih kecil
(	Kurung buka	>	Lebih besar
/	Garing		

- Aplikasi ini mampu mengubah karakter dan simbol-simbol dibawah ini sesuai dengan database yang dibuat dan keterbatasan yang dimiliki database dari Tukang Omong PD V.1.03.0020 :

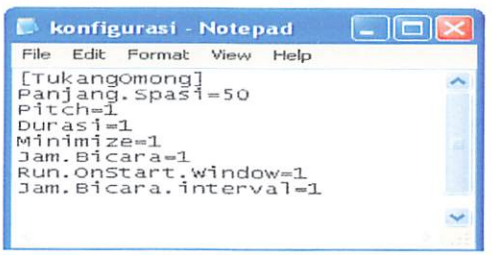
- **Konverter akronim / singkatan**

  Aplikasi ini dengan bantuan tukang omong PD V.1.03.0020 mengubah akronim menjadi huruf tunggal atau deretan huruf alphabet yang menggambarkan cara pengucapan. Dalam database Tukang Omong PD V.1.03.0020, singkatan "dg" mampu diubah menjadi "dengan". Karena sifatnya yang sangat subyektif, maka suku kata tergantung dari kebiasaan pengirim pesan sehingga tidak bisa ditentukan secara pasti. Kondisi yang masih dapat ditangani oleh aplikasi dengan bantuan Tukang omong PD V.1.03.0020, maka teks tersebut masih dapat di konversikan oleh aplikasi.

  Bila terdapat suku kata diluar keterbatasan database yang dimiliki Tukang Omong PD V.1.03.0020, maka aplikasi ini akan memberi peringatan error.

3.1.1.2 Pembangkitan intonasi

Karena sifatnya yang sangat subyektif, intonasi dalam pembacaan pesan dengan menggunakan tukang omong PD V.1.03.0020, maka ditetapkan Sebuah konfigurasi tetap yang nanti dipakai untuk proses pembacaan aplikasi TTS ini.



Gambar 3.1 Konfigurasi tetap aplikasi

Gambar 3.2 Tampilan setting konfigurasi dalam aplikasi

3.1.2 Analisis Kebutuhan Hardware dan Software

Kebutuhan hardware dan software yang akan digunakan untuk membangun sistem tersebut antara lain :

a. Intel Pentium III - keatas
b. Microsoft Windows 9x, Me, 2000 profesional dan atau XP yang telah terinstall Borland Delphi 7.
c. Memori: 32MB (diharapkan 64Mb keatas)
d. Ada sisa harddisk secukupnya untuk Data
e. CD drive
f. VGA
g. Mouse
h. SoundCard dan Speaker
i. Handphone dan Kabel Flash / Data.
j. Komponen Xcomm dan software Tukang Omong PD V.1.03.0020
k. Microsoft Office Access 2007

3.2 Metode Perancangan

Perancangan program aplikasi menggunakan pendekatan terstruktur dengan diagram alir yang menjelaskan urut-urutan proses yang terjadi pada aplikasi, disamping itu juga terdapat penjelasan mengenai algoritma program. Hasil yang diharapkan adalah berupa sebuah aplikasi yang dapat memberitahu pengguna jika ada pesan SMS yang masuk pada ponsel pengguna dan mengubah isi pesan SMS menjadi ucapan berbahasa Indonesia.

Perancangan sistem merupakan tahapan yang penting dalam pembuatan program. Perancangan bertujuan agar dalam pembuatannya dapat berjalan secara sistematis, terstruktur, dan rapi sehingga hasil program dapat optimal dan berjalan sesuai dengan apa yang dikehendaki. Aspek-aspek dalam perancangan yang diperhatikan meliputi kemungkinan pengembangan di masa depan, efektif dan efisien, kemampuan program, dan kemudahan untuk dipahami pengguna (user friendly) yang diwujudkan dalam tampilan grafis (Graphical User Interface).

Algoritma aplikasi Text To Speech berbahasa Indonesia sebagai pembaca pesan SMS adalah sebagai berikut:

1. Proses inisialisasi basisdata yang digunakan untuk menyimpan pesan SMS dan phonebook, konfigurasi pengaturan ponsel yang ada pada registry dan koneksi ke ponsel.
2. Menghubungkan aplikasi dengan ponsel.
3. Aplikasi memeriksa isi dari kotak masuk ponsel, setelah terhubung.
4. Jika pada kotak masuk pada ponsel ada pesan SMS baru yang masuk atau SMS baru yang belum terbaca maka proses pemeriksaan aplikasi berjalan, dengan

mengeluarkan pesan suara" ada SMS baru" jika ditekan Enter atau Space maka pesan akan diproses.

5. Proses SMS baru yang masuk dilakukan dengan melakukan pemisahan bagian-bagian SMS seperti nama atau nomor ponsel pengirim, tanggal penerimaan SMS, isi SMS, lokasi SMS pada ponsel.
6. Aplikasi akan membacakan pesan SMS tersebut secara otomatis dengan mengirimkan string pesan ke pembangkit ucapan untuk dibacakan dan menampilkan pesan dalam bentuk teks pada tabel pesan baru.
7. Setelah pesan SMS dibacakan, aplikasi akan menyimpan pesan SMS tersebut dalam tabel basisdata dan menghapus pesan pada ponsel.

3.2.1 Perancangan Form Aplikasi

Perancangan form aplikasi ini terdiri dari tiga form yaitu: form menu utama, form about, dan form petunjuk.

a. Form menu utama

   Form menu utama terdiri dari beberapa pilihan menu yaitu Konfigurasi , List view phone, TabSheet SMS masuk, TapSheet Phonebook, About, Petunjuk.

b. Form about

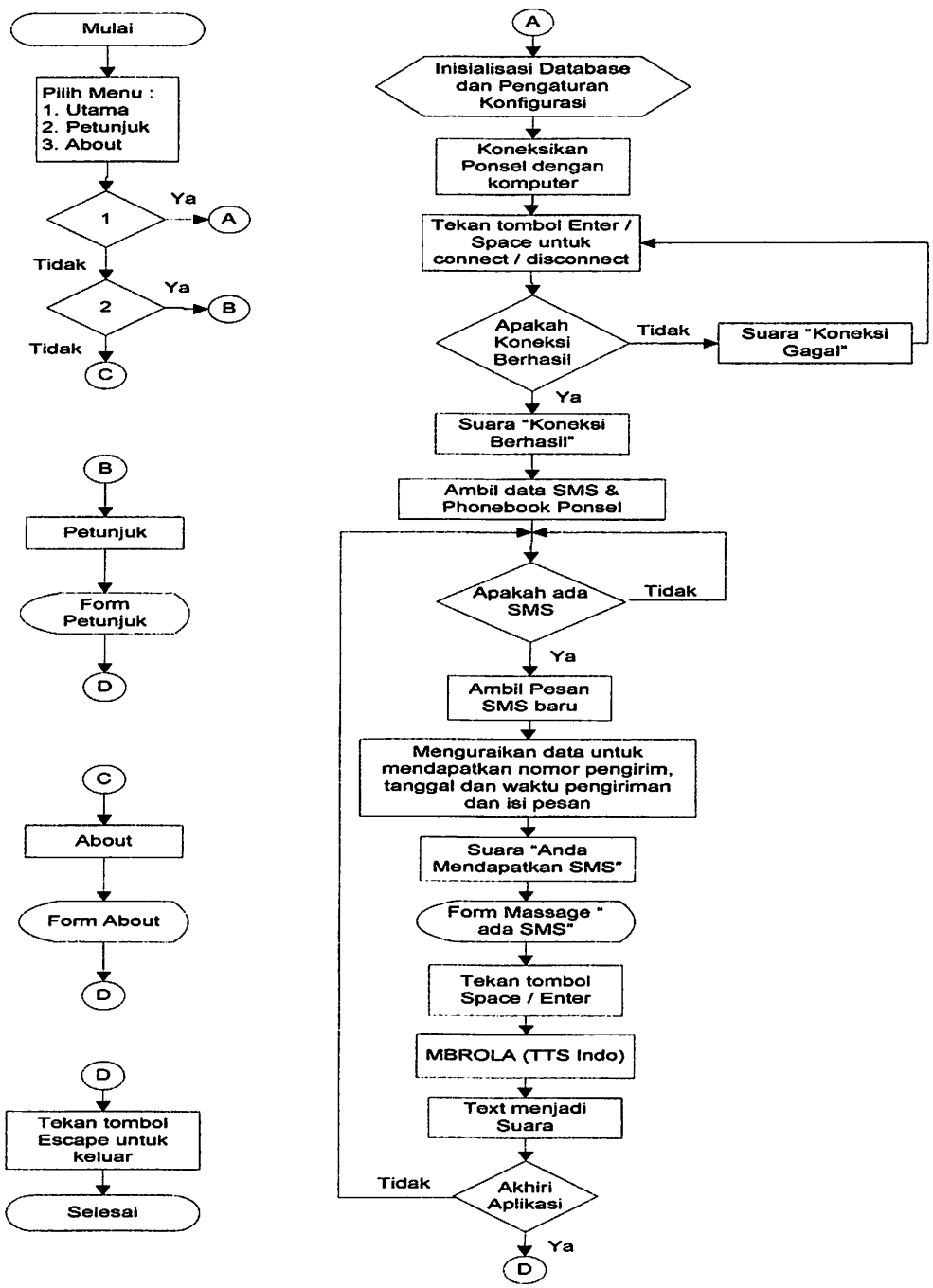
   Form tentang menampilkan informasi mengenai judul Tugas Akhir dan fungsi pembuatan aplikasi.

c. Form petunjuk

   Form petunjuk berisi tentang petunjuk penggunaan aplikasi Tugas Akhir.

3.2.2 Perancangan diagram alir

Diagram alir ini akan menggambarkan urutan penggunaan program dengan menu-menu yang tersedia.



Gambar 3.3 Diagram alir menu utama

Penjelasan dari diagram diatas adalah dalam satu aplikasi terdapat tiga form, form yang pertama adalah menu utama yang pertama tampil dilayar yang berisi button about, button petunjuk, konfigurasi phone, table SMS, tabel nomer telepon, dan memo. Form kedua adalah form button about berisi tentang penjelasan tentang program ini. Form ketiga adalah form petunjuk berisi tentang tata cara penggunaan. Jika button "Connect" pada aplikasi atau tombol space pada keyboard ditekan maka akan memerintahkan untuk melakukan koneksi antara komputer dengan ponsel dan jika konfigurasi benar maka akan mengeluarkan suatu suara tanda terkoneksi, Jika koneksi dengan ponsel sudah terbentuk, aplikasi akan memberikan informasi tentang ponsel yang digunakan dan akan memeriksa isi kotak masuk. Jika ada pesan SMS baru atau pesan SMS yang belum terbaca, pesan SMS tersebut akan diambil oleh aplikasi. Pesan SMS yang telah diambil tadi kemudian datanya dipisah-pisahkan dan diletakkan sesuai dengan tabel pesan baru yang telah tersedia, kemudian ditampilkan. Aplikasi akan memberitahu pengguna bahwa ada pesan SMS baru yang masuk, dengan memunculkan form pesan dan suara bahwa ada SMS yang masuk dan jika di tekan Space atau Enter maka text SMS yang berupa string tersebut dikirimkan ke pembangkit ucapan yang nantinya akan di ucapkan. Untuk keluar dari aplikasi harus di diskonekkan terlebih dahulu dengan cara menekan Space, setelah diskonek tekan Escape untuk menutup aplikasi.

3.2.3 Perancangan Struktur Basis Data (Database)

Database yang digunakan dalam aplikasi ini adalah Microsoft Office Access 2007 dengan pertimbangan kemudahan dalam penggunaan dan sudah dibawakan dalam satu paket Microsoft Office 2007, Perancangan basis data bukan hanya sekedar menyusun file yang diperlukan untuk disimpan sebagai basis

data, tetapi juga termasuk didalamnya bagaimana mengatur agar basis data tersebut dapat dimanfaatkan secara optimal oleh pemakai untuk memenuhi kebutuhan datanya Pengembangan sistem basis data meliputi pengembangan file basis data, perangkat lunak (software), perangkat keras (hardware)

Basis data (database) merupakan salah satu komponen yang penting dalam sistem informasi karena berfungsi sebagai penyedia informasi bagi para pemakainya. Perancangan struktur database tidak terlepas dari perancangan masukan (input) dan keluaran (output), perancangan input dan output akan berpengaruh besar terhadap kelengkapan informasi yang akan dibuat. Berikut adalah gambaran dari rancangan tabel yang akan dibuat untuk memenuhi kebutuhan database aplikasi yang akan dibuat nantinya.

3.2.3.1 Tabel Inbox

Tabel Inbox berisi tentang isi pesan dari SMS yang diterima, dengan elemen sebagai berikut :

Tabel 3.2 Tabel inbox

Key	Nama field	Type	Field size
*	NO	AutoNumber	
	Notelp_nama	Text	20
	Tanggal	Text	23
	Isi	Text	200
	Status	Text	10

*: Primary Key (kunci primer)

3.2.3.2 Tabel Phonebook

Tabel phonebook berisi tentang daftar nama dan nomor dalam phonebook, dengan elemen sebagi berikut :

Tabel 3.3 Tabel phonebook

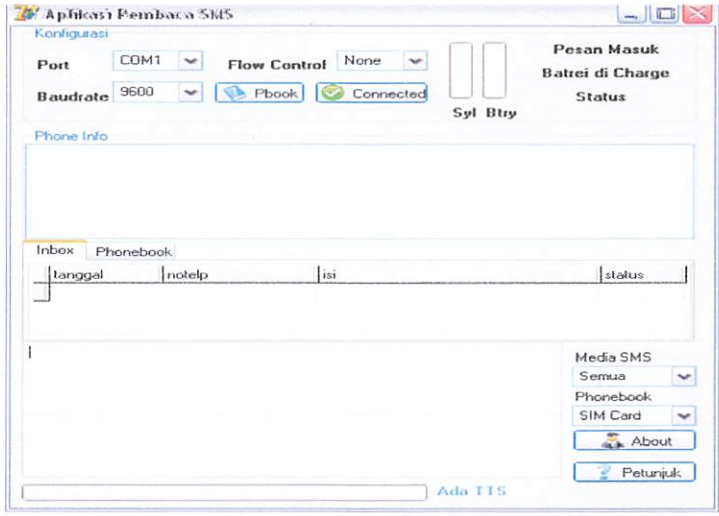
Key	Nama Field	Type	Field size
*	No	AutoNumber	
	Nama	Text	100
	Notelp	Text	25
	Jenis	Text	10

*: Primary Key (kunci primer)

3.2.4 Perancangan Antar Muka

Perancangan interface merupakan rancang bangun dari interaksi pemain sistem (administrasi) dengan komputer. Interaksi ini dapat berupa proses penginputan data ke sistem, pengupdatean data dan menjalankan aplikasi. Dalam mengimplementasikan sistem ini, penulis menggunakan satu form utama berisi beberapa alternatif atau pilihan tombol–tombol, untuk mengkoneksikan antar divice HP dan Computer, melihat form About dan Petunjuk, didalam form utama ini terdapat tampilan Inbox SMS dan Phonebook yang ada dalam handphone dan form lainnya hanya digunakan sebagai pendukung tambahan dalam aplikasi tesebut

➢ Tampilan Form Utama



Gambar 3.4 Rancangan Form Utama

Di Form inilah nantinya semua proses akan dijalankan, karena dikhususkan untuk mereka yang menyandang tuna netra dan aksara maka form utama hanya dibuat satu untuk mengefisiensikan penggunaan

➢ Tampilan Form About

Gambar 3.5 Rancangan form about

Form about disini akan berisi beberapa informasi mengenai aplikasi dan sedikit informasi mengenai status si pembuat

➢ Tampilan Form Petunjuk

Gambar 3.6 Rancangan form petunjuk

Form petunjuk disini berisi tata-tata cara penggunaan aplikasi dan penjelasan lainnya yang berhubungan dengan aplikasi

BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1 Pendahuluan

Karena program aplikasi yang di buat ini di khususkan untuk mereka para penyandang tuna netra dan tuna aksara maka dalam proses pelaksanaannya program aplikasi ini harus bisa memudahkan para penggunanya misalnya dengan mengganti fungsi button pada program aplikasi dengan button keyboard pada komputer, contohnya untuk mengkoneksikan antara media handphone dengan komputer pengguna hanya menekan button space atau enter pada keyboard dan begitu juga sama halnya untuk membaca sms baru.

Bab ini akan menjelaskan mengenai implementasi program dari keadaan yang sebenarnya jika ada penyandang tuna aksara dan tuna netra yang ingin mengetahui informasi dari SMS yang diterima dan bagaimana pengujian yang harus dilakukan agar program aplikasi ini dapat berjalan dengan semestinya.

4.2 Implementasi Sistem

4.2.1 Kebutuhan Hardware dan Software

Kebutuhan hardware dan software yang digunakan untuk menjalankan sistem tersebut antara lain :

a. Seperangkat komputer dengan spesifikasi minimum :
   1) Intel Pentium 3 dengan Clock Speed 866 Mhz
   2) Sistem Operasi Microsoft Windows XP
   3) RAM 256 MB
   4) VGA 32 MB

5) Harddisk 20 GB

6) Monitor

b. Handphone Sony Ericsson T290i

c. Komponen XComm

d. Software Tukang Omong PD V.01.03.0020

e. Kabel Flash / Data

4.2.2 Tampilan Tukang Omong PD V.01.03.0020

Tukang Omong PD V.01.03.0020 ini digunakan sebagai software pembantu untuk untuk mengkonverter teks menjadi sebuah ucapan, aplikasi ini akan membacakan string yang diterimanya oleh aplikasi yang dibuat dan nantinya akan dirubah menjadi ucapan.

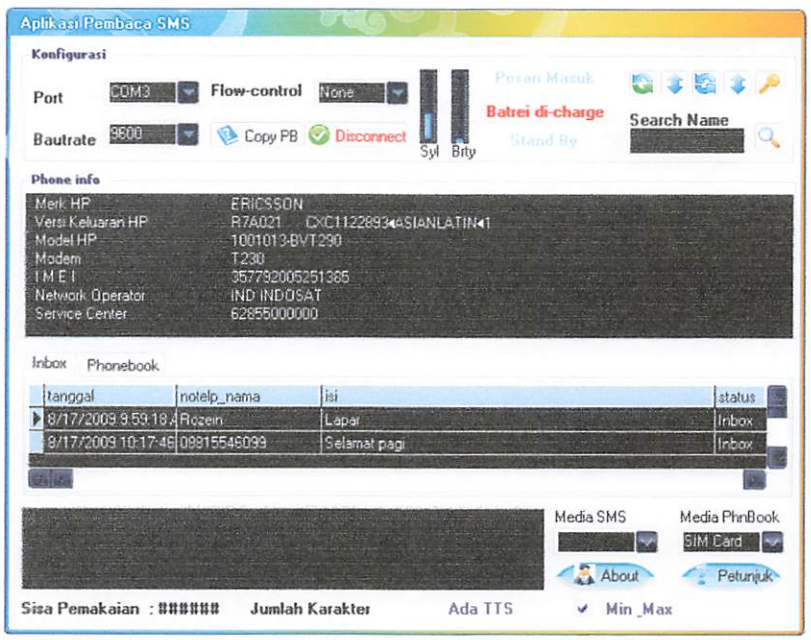
Gambar 4.1 Tampilan Tukang Omong PD V.01.03.0020 milik Lury Darmawan

4.2.3 Form Utama

Program aplikasi ini akan di ikut sertakan dengan startup jadi aplikasi ini akan terus aktif asalkan tidak di close ada aplikasi ini akan memanggil pembangkit ucapan (Tukang Omong PD) yang berada di C:\app\Tukang.Omong.PD\TukangOmong.exe jadi usahakan aplikasi pembangkit ucapan ini sudah terinstall dan berada di drive C, jika tidak ada pembangkit ucapan atau tempat installnya bukan di C:\app\Tukang.Omong.PD\ maka program

aplikasi ini akan mengeluarkan suara peringatan bahwa pembangkit ucapannya tidak ada.

Proses pertama kali untuk menggunakan aplikasi ini adalah setting konfigurasi berupa Port, Baudrate, Flow control yang nantinya akan disimpan keregistri saat aplikasi ditutup dan saat aplikasi dijalankan dia akan mengambil konfigurasi yang terakhir kali di simpan keregistri, jadi tidak perlu lagi mengeset ulang konfigurasinya asalkan sesuai dengan konfigurasi yang terakhir, kemudian melakukan koneksi antara device handphone dengan computer dengan menekan tombol enter atau space pada keyboard yang mempunyai fungsi sama dengan button " Connect ", lalu jika koneksi antar handphone dan komputer berhasil maka akan mengeluarkan suara yang menandakan koneksi berhasil ada akan muncul beberapa informasi tentang handphone yang anda gunakan pada phone info.



Gambar 4.2 Form Utama

4.2.3.1 TapSheet Phonebook

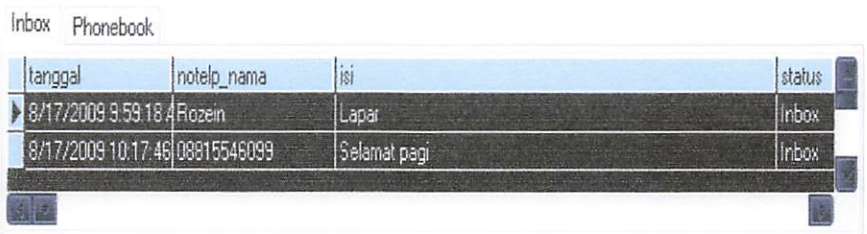
Copykan isi dari phonebook yang ada pada handphone kedatabase agar sewaktu mendapatkan SMS nama dari pengirim bisa terbaca jika tidak maka sewaktu menerima SMS hanya akan disebutkan nomer dari pengirimnya saja, cara copy phonebook dengan menekan button yang bergambarkan buku ada untuk menghapus data yang berada di database pada table phonebook gunakan button delete all phonebook yang berada disebelah button copy phonebook

Gambar 4.3 Tapsheet Phonebook

4.2.3.2 TapSheet Inbox

Prinsip untuk mengetahui SMS masuk handphone adalah dengan mengirimkan perintah AT Comand yang disisipkan ke dalam timer agar setiap saat dapat mengetahui status SMS masuk ada atau tidak, dengan mengirimkan perintah AT Comand AT+CIND? maka akan mendapatkan status dari handphone contohnya seperti ini +CIND:5,5,0,1,1,0,0,0,0,0 maksud dari angka – angka yang diterima itu adalah ('batterycharge','signal','batterywarning','chargerconnected','service','sounder','message','call','roam','callsetup') jika dibagian pesan mendapatkan logika 1 maka status seperti ini menyatakan bahwa ada SMS dan jika status berlogika 0 menyatakan bahwa tidak ada SMS. Dalam program aplikasi ini jika ada pesan masuk maka akan memberitahukan pengguna dengan mengeluarakan pesan

suara " ada pesam baru " dan akan muncul form informasi bahwa ada sms baru, untuk mendengarkan isi SMS tersebut anda bisa menggunakan tombol space atau enter pada keyboard, kemudian aplikasi akan melakukan proses pembagian data seperti nomor pengirin, tanggal, waktu pengiriman, isi pesan dari SMS yang diterima dan akan mengirimkan string pesan yang berada di dalam Memo ke pembangkit ucapan. Jika ingin mendengarkan kembali SMS yang baru di terima karena kurang jelas anda bisa gunakan tombol tanda panah atas, bawah, kanan, kiri untuk membacanya kembali. Intuk menghapus isi dari table inbox yang berada di database anda bisa gunakan button  delete sms inbox



Gambar 4.4 TapSheet Inbox

Gambar 4.5 Memo isi Sms

Gambar 4.6 Form Informasi

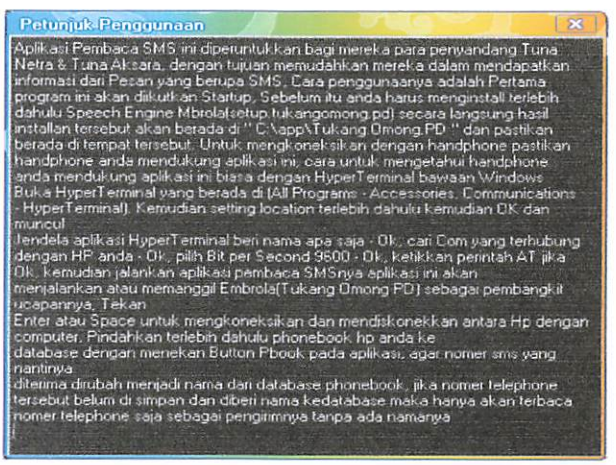
4.2.3.3 Form About

Form about disini akan berisi beberapa informasi mengenai aplikasi dan sedikit informasi mengenai profil si pembuat

Gambar 4.7 Form About

4.2.3.4 Form Petunjuk

Form petunjuk disini berisi tata-tata cara penggunaan aplikasi dan penjelasan lainnya yang berhubungan dengan aplikasi



Gambar 4.8 Form Petunjuk

4.3 Pengujian sistem

Pengujian yang dilakukan meliputi dua tahap yaitu pengujian koneksi antara ponsel dengan port USB pada komputer (digunakan kabel data), pengujian pengiriman string pesan SMS ke Tukang Omong PD. Pengujian koneksi antara ponsel dengan port USB pada komputer dilakukan dengan mengirimkan perintah AT dari program ke ponsel. Perintah AT yang digunakan adalah ATE1, apabila koneksi berhasil dilakukan ponsel akan memberikan respon "OK". Pengujian pengiriman string pesan SMS dilakukan dengan uji dengar dan mengtahui hal – hal apa saja yang menyebabkan tidak terdengarnya suatu pesan.

4.3.1 Pengujian koneksi antara ponsel dengan port USB pada komputer

Pengujian koneksi ponsel dengan port USB pada komputer dilakukan dengan mengirimkan perintah AT dari program ke ponsel dan jika ponsel akan memberikan respon "OK" maka koneksi berhasi. Ponsel akan memberikan respon "ERROR" jika koneksi gagal dilakukan. Perintah AT yang digunakan ATE1. Pada pengujian ini menggunakan ponsel Sony Ericcsion T290i yang memiliki baudrate sebesar 9600 bps. Kabel data yang digunakan untuk menghubungkan ponsel dengan komputer disambungkan dengan port USB komputer. Untuk pengujian, port yang digunakan adalah port COM 3 dan COM 4.

Komponen XComm digunakan untuk koneksi antara ponsel dengan komputer. Komponen XComm ini yang digunakan untuk mengirimkan perintah AT dari program ke ponsel. Uji komunikasi antara komputer dengan ponsel dilakukan sebanyak 10 kali. Tabel 4.1 menunjukkan hasil uji komunikasi port USB komputer dengan ponsel

Tabel 4.1 Hasil uji komunikasi antara komputer dengan ponsel.

No	*Port*	Respon	Komunikasi
1	COM 3	OK	Berhasil
2	COM 3	OK	Berhasil
3	COM 3	OK	Berhasil
4	COM 3	ERROR	Gagal
5	COM 3	ERROR	Gagal
6	COM 4	OK	Berhasil
7	COM 4	OK	Berhasil
8	COM 4	OK	Berhasil
9	COM 4	OK	Berhasil
10	COM 4	OK	Berhasil

Prosentase keberhasilan komunikasi dari prosedur komunikasi USB adalah :

$$Prosentase\ Keberhasilan = \frac{Jumlah\ percobaan\ yang\ berhasil}{Jumlah\ percobaan} \times 100\%$$

$$= \frac{8}{10} \times 100\% = 80\%$$

Dari data hasil pengujian koneksi ponsel dengan komputer dapat dikatakan bahwa program yang dibuat memiliki kinerja yang baik. Hal ini ditunjukkan dengan didapatkannya prosentase keberhasilan sebesar 80 %. Koneksi yang tidak berhasil disebabkan pada saat pengujian dilakukan, ponsel dalam dalam keadaan error atau kondisi dari ponsel tidak mendapatkan sinyal dari jaringan operator seluler.

4.3.2 Pengujian Pengiriman String Pesan Ke Tukang Omong PD

Pengujian ini dilakukan dengan mengirimkan string pesan ke Tukang Omong PD oleh aplikasi, dilakukan uji dengar apakah string pesan yang dikirimkan oleh aplikasi dapat dibaca oleh Tukang Omong PD. Pengujian dilakukan dengan mengirimkan string pesan sebanyak 10 kali ke Tukang Omong PD, string pesan SMS yang dikirimkan berupa nomor atau nama pengirim, tanggal penerimaan pesan oleh SMSC, dan isi pesan.

Tabel 4.2 Hasil pengujian pengiriman string pesan ke Tukang Omong PD

No	*String* Pesan			Uji Dengar
	Nomor/Nama Pengirim	Tanggal Penerimaan	Isi Pesan	
1	085234123699	03/08/2009 19:15:22	Makan ikan dll	Terdengar jelas
2	Rozein smart	03/08/2009 22:02:15	km skr dmn	Kurang jelas
3	Endang smart	04/08/2009 08:43:20	Lagi ngapain coy?	Terdengar jelas
4	08816235966	04/08/2009 09:40:03	Kamu sekarang dimana ?	Terdengar jelas
5	085259157799	04/08/2009 12:23:25	Jgn Lupa bawakan bajunya	Terdengar jelas
6	085234123699	05/08/2009 08:18:55	Tes ting 1 2 3 4 5 6 7 8 9	Terdengar jelas
7	Endang smart	05/08/2009 09:21:35	harganya 10000 an	Terdengar jelas
8	085234123699	05/08/2009 09:36:24	pusing nih msk gak jalan	Kurang jelas
9	Rozein smart	06/08/2009 10:35:45	gimana kbr km ?	Kurang jelas
10	Endang smart	07/08/2009 11:00:13	Alhamdulillah selesai	Terdengar jelas

Pada Tabel 4.2 dapat dilihat hasil pengujian pengiriman string pesan ke Tukang Omong PD menunjukkan tingkat keberhasilan yang baik. Informasi berupa nomor atau nama pengirim, tanggal penerimaan pesan oleh SMSC, dan isi pesan dapat diterima oleh Tukang Omong PD dan terdengar sesuai dengan yang dikirim oleh aplikasi. Kurang jelasnya suatu pengucapan yang terjadi disebabkan oleh keterbatasan dari database yang dimiliki oleh aplikasi Tukang Omong PD contoh dari beberapa faktor menyebabkan tidak jelasnya suatu pengucapan yang dilakukan oleh aplikasi Tukang Omong DP seperti penggunaan kata - kata yang tidak baku dari bahasa Indonesia, kata – kata singkatan yang tidak dimengerti oleh pembangkit ucapan, dan ada beberapa hal yang bisa menyebabkan terjadinya suatu error pada pembangkit ucapan sehingga tidak akan mengeluarkan suara, pertama jangan menggunakan kata atau kalimat yang didalamnya disisipi dengan huruf vocal ' x, q ' dan perulangan huruf vocal seperti aa, bb , dan yang kedua jika akan mengirim pesan dengan menggunakan symbol atau karakter khusus maka harap dipisah dengan space.

BAB V

PENUTUP

5.1 KESIMPULAN

Dari perancangan dan implementasi yang telah dilakukan ada beberapa kesimpulan yang diperoleh berdasarkan " Pengembangan Aplikasi Text To Speech Bahasa Indonesia Sebagai Pembaca SMS Menggunakan Borland Delphi 7 " adalah sebagai berikut :

1. Aplikasi Text To Speech Berbahasa Indonesia sebagai Pembaca SMS ini dapat digunakan sebagai salah satu alternatif untuk membaca SMS yang ada pada ponsel GSM, menyimpannya dan merubah SMS tadi menjadi ucapan. Dengan menggunakan metode pengambilan pesan SMS pada ponsel melalui komputer, yang selanjutnya dengan bantuan software Tukang Omong PD V.1.03.0020 mengubah teks pesan SMS tersebut menjadi ucapan.
2. Aplikasi Text To Speech Berbahasa Indonesia sebagai Pembaca SMS ini dapat memberikan kemudahan kepada pengguna dalam mendapatkan informasi pesan teks SMS, tanpa perlu intensitas tinggi untuk memahami sebuah pesan yang masuk pada ponsel.
3. Pengujian aplikasi dengan ponsel Sony Ericsson T290i menunjukkan persentase keberhasilan komunikasi sebesar 80%.
4. Pengiriman string pesan singkatan ke aplikasi TTS dapat terbaca dan terdengar oleh Tukang Omong PD bila sesuai dengan database yang ada, jadi agar terbaca dengan jelas maka pesan yang akan dikirim harus sesuai dengan standart baku dari kata - kata bahasa Indonesia yang baik dan benar, jika tidak maka tidak akan bisa dibaca oleh Tukang Omong PD atau terjadi error karena keterbatasan database.

5.2 SARAN

Disadari sistem ini masih banyak kekurangan dan kelemahan, oleh karena itu untuk pengembangan selanjutnya disarankan :

1. Perlu dilakukan penelitian dengan ponsel jenis lain agar sistem tidak tergantung hanya dengan satu jenis ponsel saja.
2. karena setiap manusia mempunyai sifat yang sangat subyektif, maka perlu dibuat sebuah database untuk normalisasi teks dan pembangkitan intonasi yang lebih kompleks.

Demikian saran yang dapat diberikan penulis sebagai bahan masukan agar aplikasi berikutnya bias lebih baik dari sebelumnya

.

DAFTAR PUSTAKA

[1] Indra , Jurnal Dasar SMS gateway, Mobile Programming, Fakultas Teknologi Informasi Universitas Budi Luhur.

[2] http://www.ittelkom.ac.id/library/index.php?view=article&catid=17%3Asistem-komunikasi-bergerak&id=411%3Asms-short-message-service&tmpl=component&print=1&page=&option=com_content&Itemid=15, SMS (Short Message Service), Written by admin Tuesday, 10 February 2009 06:28.

[3] http://www.informatika.org/~rinaldi/Matdis/2007-2008/Makalah/MakalahIF2153-0708-086.pdf, Nadhira Ayuningtyas.

[4] http://pranedya_pratama.students-blog.undip.ac.id/category/telekomunikasi/, Pranedya's

[5] http://www.globalkomputer.com/Bahasan/Komunikasi-Data/Topik/PDU/Subtopik/SMS-Submit-PDU.html, SMS Submit PDU

[6] Jurnal Rancangan Dan Implementasi Prototipe Sistem Kendali Jarak Jauh Berbasis AT89C52 Dan Layanan SMS GSM, Jazi Eko Istiyanto dan Yeyen Efendy, 2004

[7] http://www.Bengkelprogram.com, Aplikasi Handphone, Aryo Sanjaya, 2005-2009.

[8] http://www.Bengkelprogram.com, Membaca SMS dari PC, Aryo Sanjaya, 2005-2009.

[9] http://ejournal.unud.ac.id/abstrak/putra%20sastra_5_.pdf, Perancangan dan pembuatan system kontrol dengan memanfaatkan layanan SMS telepon seluler berbasis mikrokontroler AT89C51, I Nyoman Putra Sastra, Dewa Made Wiharta, Agus Supranartha, Desember 2005

[10] http://www.wikipedia.com, ASCII

[11] http://komputer-kandagalante.blogspot.com/2008/09/sms.html, SMS, Yeni Fatimah, 2008

[12] http://www.warnaku.com/Konsep pembuatan sms gateway_warna warni.html, Konsep pembuatan sms gateway_warna warni, 2007

[13] http://www.pustaka-kita.com/Tutorial SMS gateWay.html, Tutorial SMS gateway, Andy, 2009

[14] http://indotts.melsa.net.id/, Arry Akhmad Arman

[15] Deny Yunus Riyadi, Aplikasi Text To Speech bahasa Indonesia sebagai Pembaca Sms, , 2007

[16] www.Kios.com, September 2004.

[17] Subali Muhammad, Makalah Model linier dinamik sebagai dasar penyeleksian DIPHONE pada sistem pensintesis suara Concatenative dalam bahas Indonesia, 2004.

[18] http://teknologibahasa.wordpress.com/,

[19] http://kepac.nomi.cz/bordylek/gsm_sms.pas,

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO S-1
Jl. Karanglo KM.2 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama : Rozein Rousyanfikri

Nim : 05.12.577

Jurusan : Teknik Elektro S-1

Konsentrasi : Teknik Komputer dan Informatika

Judul Skripsi : **PENGEMBANGAN APLIKASI TEXT TO SPEECH BERBAHASA INDONESIA SEBAGAI PEMBACA SMS MENGGUNAKAN BORLAND DELPHI 7**

Dipertahankan di hadapan Majelis Penguji Skripsi Jenjang Strata Satu (S-1) pada :

Hari : Senin

Tanggal : 07 September 2009

Dengan Nilai : 88,05 (A)

Ketua Majelis Penguji

Ir. H. Sidik Noertjahjono, MT
NIP.Y. 1028700163

Sekretaris Majelis Penguji

Ir. F. Yudi Limpraptono, MT
NIP Y. 1039500274

Penguji I

Ir.Th.Mimien Mustikawati, MT
NIP.P 1030000352

Penguji II

Sandy Nataly M S.KOM

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura – gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Rozein Rousyanfikri
Nim : 05.12.577
Masa Bimbingan : 22-Juni-2009 s/d 22-Desember-2009
Judul Skripsi : Pengembangan Aplikasi Text To Speech Bahasa Indonesia Sebagai Pembaca SMS Menggunakan Borland Delphi 7.

No	Tanggal	Uraian	Paraf Pembimbing
1	4/8/09	Bab I – III revisi	
2	7/8/09	Bab IV & V revisi	
3	11/8/09	Gambar lebih besar	
4		Demo	
5		Seminar hasil	
6	2/9/09	Laporan lengkap (diperbaiki	
7			
8			
9			
10			

Malang,

Dosen pembimbing I

Ir.F.Yudi Limpraptono.MT
NIP. Y. 1039500274

Form S-4b

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Janjang Strata 1 Jurusan Teknik Elektro Konsentrasi T. Energi Listrik / T. Elektronika / T.Infokom, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : LUZEIN ROUSYANFIKR
NIM : 0512577
Perbaikan meliputi :

BUAT BLOK DIAGRAM YG BENAR

Malang, 7 SEPT 2009

(SAMPAI)

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK KOMPUTER & INFORMATIKA S-1
Jl. Karanglo KM. 2 Malang

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan Ujian Skripsi Jenjang Strata Satu (S-1) Jurusan Teknik Elektro Konsentrasi Teknik Komputer & Informatika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

Nama	: Rozein Rousyanfikri
NIM	: 05 12 577
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Komputer & Informatika
Masa Bimbingan	: 22 Juni 2009 s/d 22 Desember 2009
Judul Skripsi	: Pengembangan Aplikasi Text To Speech Berbahasa Indonesia Sebagai Pembaca SMS Menggunakan Borland Delphi 7

Penguji/Tanggal	Uraian	Paraf
Penguji I 7 September 2009	Buat blog diagram yang benar	

Mengetahui

Dosen Penguji I

(Sandy Nataly M, S.KOM)

Dosen Pembimbing I

(Ir. F Yudi Limpraptono, MT)
NIP.Y. 1039500274

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, Buttons, StdCtrls, ComCtrls, ExtCtrls, XComDrv, Registry,
XPMan,
  Grids, DBGrids, DB, gsm_sms, ADODB, sSkinManager, sSkinProvider,
sBitBtn;

type
  TForm1 = class(TForm)
    XComm1: TXComm;
    Timer1: TTimer;
    XPManifest1: TXPManifest;
    GroupBox1: TGroupBox;
    Label1: TLabel;
    Label2: TLabel;
    SpeedButton1: TSpeedButton;
    Label3: TLabel;
    Label4: TLabel;
    ComboBoxPort: TComboBox;
    ComboBoxBaut: TComboBox;
    ProgressBarSinyal: TProgressBar;
    ProgressBarBattery: TProgressBar;
    ComboBoxFc: TComboBox;
    Label5: TLabel;
    DataSource1: TDataSource;
    ListViewSms: TListView;
    ComboBoxSumber: TComboBox;
    Label6: TLabel;
    ADOConnection1: TADOConnection;
    ADOTable1: TADOTable;
    ADODataSet1: TADODataSet;
    indSMSBaru: TLabel;
    indCharged: TLabel;
    indStatus: TLabel;
    Memo2: TMemo;
    TimerCheckStatusSMS: TTimer;
    GroupBox2: TGroupBox;
    ListViewPhone: TListView;
    LabelSisaSMS: TLabel;
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    DBGrid1: TDBGrid;
    LabelStatusTTS: TLabel;
    ComboBoxPbook: TComboBox;
    TabSheet2: TTabSheet;
    DBGrid2: TDBGrid;
    ADODataSet2: TADODataSet;
    ADOTable2: TADOTable;
    DataSource2: TDataSource;
    Label7: TLabel;
    SpeedButton4: TSpeedButton;
    ProgressBar1: TProgressBar;
    TimerStatusTTS: TTimer;
    SpeedButton5: TSpeedButton;
    Labelsisa: TLabel;
    Labelkey: TLabel;
    SpeedButton6: TSpeedButton;
    CheckBox1: TCheckBox;
    Labelmax: TLabel;
    SpeedButton7: TSpeedButton;
    ADOQuery1: TADOQuery;
```

```
    Edit1: TEdit;
    Label8: TLabel;
    SpeedButton8: TSpeedButton;
    SpeedButton9: TSpeedButton;
    SpeedButton10: TSpeedButton;
    ADOTable1no: TAutoIncField;
    ADOTable1notelp_nama: TWideStringField;
    ADOTable1tanggal: TWideStringField;
    ADOTable1isi: TWideStringField;
    ADOTable1status: TWideStringField;
    sSkinManager1: TsSkinManager;
    sBitBtn1: TsBitBtn;
    sBitBtn2: TsBitBtn;
    procedure FormCreate(Sender: TObject);
    procedure XComm1CommEvent(Sender: TObject; const Events:
TDeviceEvents);
    procedure SpeedButton1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure DBGrid1CellClick(Column: TColumn);
    procedure TimerCheckStatusSMSTimer(Sender: TObject);
    procedure Memo2KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure Memo2Change(Sender: TObject);
    procedure ListViewPhoneKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure SpeedButton4Click(Sender: TObject);
    procedure TimerStatusTTSTimer(Sender: TObject);
    procedure SpeedButton5Click(Sender: TObject);
    procedure SpeedButton6Click(Sender: TObject);
    procedure CheckBox1Click(Sender: TObject);
    procedure SpeedButton7Click(Sender: TObject);
    procedure FormDestroy(Sender: TObject);
    procedure ComboBoxPortKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure DBGrid1KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure DBGrid2KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure ComboBoxPbookKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure ComboBoxSumberKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure CheckBox1KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure ComboBoxBautKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure ComboBoxFcKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure FormKeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure Memo1KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure Edit1KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure SpeedButton8Click(Sender: TObject);
    procedure SpeedButton9Click(Sender: TObject);
    procedure SpeedButton10Click(Sender: TObject);
    procedure sBitBtn1Click(Sender: TObject);
    procedure sBitBtn2Click(Sender: TObject);

  private
    { Private declarations }
    function GoKonek: Boolean;
    procedure SetTerkoneksi;
    procedure getInfo(buffer: string);
  public
```

```
    { Public declarations }
    function SendGetData(Teks, Batas: String): String;
    procedure SimpanSetting;
    procedure BacaSetting;
    procedure CekSMS;
    procedure SimpanSMS;
    procedure Delay(Lama : Longint);

  end;

const   sOK = #13#10'OK';
        sERROR = #13#10'ERROR';

        arSumber: array[0..2] of String[2] = ('SM', 'ME', 'MT');
        arPhoneInfo: array[1..10] of string[20] = ('battchg',
'signal', 'batterywarning',
                                                   'chargerconnected',
'service', 'sounder',
                                                   'message', 'call',
'roam', 'callsetup');
var
  Form1: TForm1;
  Konek,SedangBaca,ReadyState: Boolean;
  Item,List,List2: TStrings;

  Buffer: WideString;
  sYangMoDiOmongin,BatasStr,SMSC: String;

  PhoneInfo : array[1..20] of string;
  JumlahPhoneInfo: integer;

implementation

uses StrUtils, DateUtils, MMSystem,pustakaUserSpeechInterface ,
Unit2,
  Unit3, Unit4;

{$R *.dfm}

//
========================Procedure&Function=========================
======
procedure TForm1.Delay(Lama : Longint);
  var ref : Longint;
begin
   ref := GetTickCount;
   repeat
      Application.ProcessMessages;
   Until (( GetTickCount-ref)>= Lama) ;
end;

function TrimAll(t: string): string;
  var s: string;
begin
  s := trim(t);
  s := copy(s, 2, length(s) - 2);
  result := s;
end;

function TForm1.SendGetData;
  var waktu: TDateTime;
begin
  ReadyState := False;
  BatasStr := Batas;
  Buffer := '';
  waktu := now;
```

```
  XComm1.SendString(Teks);
  while (Not ReadyState) and (SecondsBetween(waktu, Now) < 10)do
Application.ProcessMessages;
    Result := Buffer;
end;

function TForm1.GoKonek;
begin
  Konek := false;
     If Not Form1.XComm1.Opened then
        Form1.XComm1.OpenDevice;
     if XComm1.SendString('ATE1'#13) and (XComm1.WaitForString
(['OK'], 2000) <> -1) then
        Konek := True;
     Result := Konek;
end;

procedure TForm1.SimpanSetting;
  var Reg : TRegistry;
begin
  Reg := TRegistry.Create;
  try
    if Reg.OpenKey('\Software\Rozein\HPApp', True) then
   begin
      Reg.WriteInteger('BaudRate', ComboBoxBaut.ItemIndex);
      Reg.WriteInteger('Port', ComboBoxPort.ItemIndex);
      Reg.WriteInteger('Flowcontrol', ComboBoxFc.ItemIndex);
      Reg.WriteString('TryKey', Labelkey.Caption);
      Reg.CloseKey;
    end;
  finally
    Reg.Free;
    inherited;
  end;
end;

procedure TForm1.SetTerkoneksi;
begin
  If not Konek then Exit;
    ListViewPhone.Items.Clear;
    ADOConnection1.Connected:=true;

    getInfo(SendGetData('AT+CGMI'#13, sOK));
    getInfo(SendGetData('AT+CGMR'#13, sOK));
    getInfo(SendGetData('AT+CGMM'#13, sOK));
    getInfo(SendGetData('AT+GMM'#13, sOK));
    getInfo(SendGetData('AT+CGSN'#13, sOK));
    getInfo(SendGetData('AT+COPS?'#13, sOK));
    getInfo(SendGetData('AT+CSCA?'#13, sOK));
    getInfo(SendGetData('AT+CIND=?'#13, sOK));

    Timer1.Enabled := Konek;
    ADODataSet1.Active:=True;
    ADODataSet2.Active:=True;

    SpeedButton1.Caption := 'Disconnect';
    SpeedButton1.Font.Color:=clRed;
    ComboBoxPort.Enabled := False;
    ComboBoxBaut.Enabled := False;
    ComboBoxFc.Enabled := False;
    SedangBaca := False;
    SimpanSetting;
    sYangMoDiomongin:='Koneksi berhasil';
    omongin(sYangMoDiomongin);
    sndPlaySound('Sound/Konek.wav',SND_FILENAME);
end;
```

```
procedure TForm1.getInfo;
  var c,s: string;
      p,n,i: integer;
      l: TListItem;
begin
  List.Text := buffer;
  s := List.Strings[1];
  if Pos('AT+CGMI', Buffer) > 0 then begin
    c := copy(Buffer, pos('AT+CGMI', Buffer) + 2, length(Buffer));
    Item.Text := c;
    l := ListViewPhone.Items.Add;
    l.Caption := 'Merk HP';
    l.SubItems.Add(Item.Strings[2]);
  end;
  if Pos('AT+CGMR', Buffer) > 0 then begin
    c := copy(Buffer, pos('AT+CGMR', Buffer) + 2, length(Buffer));
    Item.Text := c;
    l := ListViewPhone.Items.Add;
    l.Caption := 'Versi Keluaran HP';
    l.SubItems.Add(Item.Strings[2]);
  end;
  if Pos('AT+CGMM', Buffer) > 0 then begin
    c := copy(Buffer, pos('AT+CGMM', Buffer) + 2, length(Buffer));
    Item.Text := c;
    l := ListViewPhone.Items.Add;
    l.Caption := 'Model HP';
    l.SubItems.Add(Item.Strings[2]);
  end;
  if Pos('AT+GMM', Buffer) > 0 then begin
    c := copy(Buffer, pos('AT+GMM', Buffer) + 2, length(Buffer));
    Item.Text := c;
    l := ListViewPhone.Items.Add;
    l.Caption := 'Modem';
    l.SubItems.Add(Item.Strings[2]);
  end;
  if Pos('AT+CGSN', Buffer) > 0 then begin
    c := copy(Buffer, pos('AT+CGSN', Buffer) + 2, length(Buffer));
    Item.Text := c;
    l := ListViewPhone.Items.Add;
    l.Caption := 'I M E I';
    l.SubItems.Add(Item.Strings[2]);
  end;
  if Pos('+COPS:', Buffer) > 0 then begin
    c := copy(Buffer, pos('+COPS:', Buffer) + 7, length(Buffer));
    c := AnsiReplaceStr(c, ',', #13);
    Item.Text := c;
    l := ListViewPhone.Items.Add;
    l.Caption := 'Network Operator';
    l.SubItems.Add(TrimAll(Item.Strings[2]));
  end;
  if Pos('+CSCA:', Buffer) > 0 then begin
    c := copy(Buffer, pos('+CSCA:', Buffer) + 7, length(Buffer));
    c := AnsiReplaceStr(c, ',', #13);
    Item.Text := c;
    l := ListViewPhone.Items.Add;
    l.Caption := 'Service Center';
    SMSC := TrimAll(Item.Strings[0]);
    l.SubItems.Add(SMSC);
  end;
  if Pos('+CIND:', Buffer) > 0 then begin
    c := copy(Buffer, pos('+CIND:', Buffer) + 7, length(Buffer));
    i := 0;
    p := pos('(', c);
    while p > 0 do begin
      n := pos('),', c);
```

```
      s := copy(c, p + 2, n - p - 2);
      PhoneInfo[i] := s;
      Inc(i);
      Delete(c, p, n + 2);
      p := pos('("', c);
    end;
    JumlahPhoneInfo := i;
  end;
end;

procedure TForm1.BacaSetting;
  var Reg : TRegistry;
begin
  Reg := TRegistry.Create;
  try
    if Reg.OpenKey('\Software\Rozein\HPApp',True) then
   begin
      if Reg.ValueExists('BaudRate') then
        ComboBoxBaut.ItemIndex := Reg.ReadInteger('BaudRate')
        else
        ComboBoxBaut.ItemIndex := 0;
      if Reg.ValueExists('Port') then
        ComboBoxPort.ItemIndex := Reg.ReadInteger('Port')
        else
        ComboBoxPort.ItemIndex := 0;
      if Reg.ValueExists('Flowcontrol') then
        ComboBoxFc.ItemIndex := Reg.ReadInteger('Flowcontrol')
        else
        ComboBoxFc.ItemIndex := 0;
      if Reg.ValueExists('TryKey') then
        Labelkey.Caption := Reg.ReadString('TryKey')
      else
        Labelkey.Caption := '0';
      Reg.CloseKey;
    end;
  finally
    Reg.Free;
    inherited;
  end;
end;

procedure TForm1.CekSMS;
  var nomer,tgl,s : string;
      i,j : integer;
      l: TListItem;
      sms : TSMS;
begin
  if not Konek then exit;
  s := sendGetData('AT+CSMS=0'#13, sOK);
  if Pos(sOK, s) = 0 then begin
    sYangMoDiomongin:='Media tidak mendukung fasilitas S M S';
    omongin(sYangMoDiomongin);
    ShowMessage('Media tidak mendukung fasilitas SMS');
    exit;
  end;
  s := sendGetData('AT+CPMS="' + arSumber[ComboBoxSumber.ItemIndex]
+'"'#13, sOK);
  if Pos(sOK, s) = 0 then begin
    sYangMoDiomongin:='Tidak dapat memilih media ' +
ComboBoxSumber.Text;
    omongin(sYangMoDiomongin);
    ShowMessage('Tidak dapat memilih media ' + ComboBoxSumber.Text);
    exit;
  end;
  s := sendGetData('AT+CMGF=0'#13, sOK);
  if Pos(sOK, s) = 0 then begin
```

```
    sYangMoDiomongin:='Tidak dapat memilih format P D U';
    omongin(sYangMoDiomongin);
    ShowMessage('Tidak dapat memilih format PDU');
    exit;
  end;
  List.Clear;
  sms := TSMS.Create;
  for j := 3 downto 0 do begin
    List.Text := SendGetData('AT+CMGL=' + IntToStr(j) +#13, sOK);
    i := 0;
    while i < List.Count do begin
      s := List.Strings[i];
      if copy(s, 1, 7) = '+CMGL: ' then begin
        nomer := copy(s, 8, pos(',', s) - 8);
        inc(i);
        s := List.Strings[i];
        sms.PDU := s;
        l := ListViewSms.Items.Add;
        if sms.TimeStamp > 0 then tgl := DateTimeToStr(sms.TimeStamp)
        else tgl := '-';
          l.SubItems.Add(tgl);
          l.SubItems.Add(sms.Text);
          l.SubItems.Add(sms.number);
          l.SubItems.Add(nomer);  //index di hp
          l.SubItems.Add(nomer);  //id sms
        end;
        inc(i);
      end;
    end;
  sms.Free;
end;

procedure TForm1.SimpanSMS;
  var i: integer;
      l: TListItem;
      str1,HSL,NMR,R,s,n: string;
begin
  Timer1.Enabled := False;
  for i := ListViewSms.Items.Count - 1 downto 0 do begin
    l := ListViewSms.Items.Item[i];
    n := l.SubItems[4];
    s := SendGetData('AT+CMGD=' + n + #13, sOK);
    if Pos(sOK, s) > 0 then begin
      NMR:='0'+copy((l.SubItems[2]),4,20);
      with ADOQuery1 do begin
        Close;
        SQL.Clear;
        SQL.add('SELECT count(*) as jml FROM phonebook WHERE notelp=
:V0');
        Parameters[0].Value:=NMR;
        Active:=true;
        Open;
        str1:=ADOQuery1['jml'];
        Active:=False;
      end;
      if strtoint(str1)>0 then
        begin
          ADOQuery1.Close;
          ADOQuery1.SQL.Clear;
          ADOQuery1.SQL.Text:='select nama from Phonebook where
notelp LIKE'+ QuotedStr('%'+NMR+ '%');
          ADOQuery1.Active:=true;
          //ADOQuery1.Open;
          HSL:=ADOQuery1['nama'];
          ADOQuery1.Active:=false;
```

```
          ADODataSet1.Active:=False;
          ADOTable1.Active:=True;
          ADOTable1.Append;
          ADOTable1['tanggal']:=l.SubItems[0];
          ADOTable1['notelp_nama']:= HSL;
          ADOTable1['isi']:=l.SubItems[1];
          ADOTable1['status']:='Inbox';
          ADOTable1.Post;
          ADOTable1.Active:=False;
          ADODataSet1.CommandText:='select
tanggal,notelp_nama,isi,status from inbox order by tanggal desc';
          ADODataSet1.Active:=True;
          Memo2.Lines.Add('dari'+' '+HSL+' '+'tanggal'+' '+l.SubItems
[0]+' '+'isi pesan adalah'+' '+l.SubItems[1]);
          TimerCheckStatusSMS.Enabled:=True;
          if MessageDlg('Ada Sms Baru',mtInformation,[mbYes],0)=
mrYes then
          begin
            TimerCheckStatusSMS.Enabled:=False;
            Omongin(Memo2.Lines.Text);
            Delay(30000);
            Memo2.Clear;
          end;
          indSMSBaru.Font.Color := clSkyBlue;
        end
      else
        if copy((l.SubItems[2]),1,1)='+' then
        begin
          ADODataSet1.Active:=False;
          ADOTable1.Active:=True;
          ADOTable1.Append;
          ADOTable1['tanggal']:=l.SubItems[0];
          ADOTable1['notelp_nama']:='0'+copy((l.SubItems[2]),4,20);
          ADOTable1['isi']:=l.SubItems[1];
          ADOTable1['status']:='Inbox';
          ADOTable1.Post;
          ADOTable1.Active:=False;
          ADODataSet1.CommandText:='select
tanggal,notelp_nama,isi,status from inbox order by tanggal desc';
          ADODataSet1.Active:=True;
          R:='0'+copy((l.SubItems[2]),4,20);
          Memo2.Lines.Add('dari'+' '+R+' '+'tanggal'+' '+l.SubItems
[0]+' '+'isi pesan adalah'+' '+l.SubItems[1]);
          TimerCheckStatusSMS.Enabled:=True;
          if MessageDlg('Ada Sms Baru',mtInformation,[mbYes],0)=
mrYes then
          begin
            TimerCheckStatusSMS.Enabled:=False;
            Omongin(Memo2.Lines.Text);
            Delay(30000);
            Memo2.Clear;
          end;
          indSMSBaru.Font.Color := clSkyBlue;
        end;
        if not (copy((l.SubItems[2]),1,1)='+') then
        begin
          ADODataSet1.Active:=False;
          ADOTable1.Active:=True;
          ADOTable1.Append;
          ADOTable1['tanggal']:=l.SubItems[0];
          ADOTable1['notelp_nama']:=l.SubItems[2];
          ADOTable1['isi']:=l.SubItems[1];
          ADOTable1['status']:='Inbox';
          ADOTable1.Post;
          ADOTable1.Active:=False;
          ADODataSet1.CommandText:='select
```

```
tanggal,notelp_nama,isi,status from inbox order by tanggal desc';
          ADODataSet1.Active:=True;
          Memo2.Lines.Add('dari'+' '+1.SubItems[2]+' '+'tanggal'+'
'+1.SubItems[0]+' '+'isi pesan adalah'+' '+1.SubItems[1]);
          TimerCheckStatusSMS.Enabled:=True;
          if MessageDlg('Ada Sms Baru',mtInformation,[mbYes],0)=
mrYes then
          begin
            TimerCheckStatusSMS.Enabled:=False;
            Omongin(Memo2.Lines.Text);
            Delay(30000);
            Memo2.Clear;
          end;
          indSMSBaru.Font.Color := clSkyBlue;
        end;

      end;
    end;
  ListViewSms.Items.Clear;
  Timer1.Enabled:=True;
end;

//
================================End================================
====
procedure TForm1.XComm1CommEvent(Sender: TObject; const Events:
TDeviceEvents);
  var data : string;
begin
  XComm1.ReadString(data);
  Buffer := Buffer + data;
    If (Not ReadyState) And (Pos(BatasStr, Buffer) > 0) Then
    Begin
      ReadyState := True;
    End;
end;

procedure TForm1.FormCreate(Sender: TObject);
  var i,k : integer;
begin
  WinExec('C:\app\Tukang.Omong.PD\TukangOmong.exe',1);
  Form1.WindowState := wsMaximized;
  Form1.FormStyle:=fsStayOnTop;
  TimerStatusTTS.Enabled:=True;

  Item := TStringList.Create;
  List := TStringList.Create;
  List2 := TStringList.Create;
  Konek := False;
  Buffer := '';
  Timer1.Enabled := False;
  ComboBoxPort.Items.Clear;
  for i := 1 to 30 do
  ComboBoxPort.Items.Add('COM' + IntToStr(i));
  ComboBoxPort.ItemIndex := 0;

  ComboBoxBaut.Items.Clear;
  ComboBoxBaut.Items.Add('9600');
  ComboBoxBaut.Items.Add('19200');
  ComboBoxBaut.Items.Add('57600');
  ComboBoxBaut.Items.Add('115200');
  ComboBoxBaut.Items.Add('128000');
  ComboBoxBaut.Items.Add('256000');
  ComboBoxBaut.ItemIndex := 0;

  ComboBoxFc.Clear;
```

```
  ComboBoxFc.Items.Add('None');
  ComboBoxFc.Items.Add('RTS-CTS');
  ComboBoxFc.Items.Add('DTR-DSR');
  ComboBoxFc.Items.Add('Software');
  ComboBoxFc.ItemIndex := 0;

  ComboBoxPbook.Clear;
  ComboBoxPbook.Items.Add('SIM Card');
  ComboBoxPbook.Items.Add('Phone');
  ComboBoxPbook.Items.Add('Semua');
  ComboBoxPbook.ItemIndex := 0;

  XComm1.BaudRate := brCustom;
  XComm1.FlowControl := fcNone;

  ListViewSms.Items.Clear;
  BacaSetting;

ADODataSet1.Active:=False;
ADODataSet1.CommandText:='select no,tanggal, notelp_nama, isi, status
from inbox';
ADODataSet2.Active:=False;
ADODataSet2.CommandText:='select nama, notelp, jenis from Phonebook';

  if Labelkey.Caption='0' then
    begin
      sYangMoDiomongin:='Silahkan aktifasi terdahulu silahkan hubungi
rozein';
      omongin(sYangMoDiomongin);
      ShowMessage('Silakan Aktifasi dahulu, Silakan Hub RZN');
      exit;
    end
  else
  if Labelkey.Caption='######' then
    begin
    //ShowMessage('Aplikasi Ini Milik RZN');
    end
  else
  begin
    k:= strtoint(Labelkey.Caption)-1;
    Labelkey.Caption:=inttostr(k);
    //ShowMessage('Aplikasi Ini Milik RZN');
    //exit;
  end;
end;

procedure TForm1.SpeedButton1Click(Sender: TObject);
begin
  if Labelkey.Caption='0' then
    begin
      sYangMoDiomongin:='Silahkan aktifasi terlebih dahulu';
      omongin(sYangMoDiomongin);
      ShowMessage('Silakan Aktifasi Terlebih dahulu');
      exit;
    end
  else

  if Labelkey.Caption='######'then begin
      if Konek then begin
        Timer1.Enabled := false;
        XComm1.CloseDevice;
        SpeedButton1.Caption := 'Connect';
        Konek := False;
        ComboBoxPort.Enabled := True;
        ComboBoxBaut.Enabled := True;
        ComboBoxFc.Enabled := True;
```

```
        ProgressBarSinyal.Position := 0;
        ProgressBarBattery.Position := 0;

        ADODataSet1.Active:=False;
        ADODataSet2.Active:=False;
        ADOConnection1.Connected:=false;
      end
    else begin
      XComm1.BaudValue := StrToInt(ComboBoxBaut.Text);
      XComm1.DeviceName := ComboBoxPort.Text;
      XComm1.FlowControl := TFlowcontrol(ComboBoxFc.ItemIndex);
      If Not GoKonek Then begin
        sYangMoDiomongin:='Gagal membuka Port'+ComboBoxPort.Text;
        omongin(sYangMoDiomongin);
        ShowMessage('Gagal membuka port');
        end
        else
        SetTerkoneksi;
      end;
  end

  else begin
    if Konek then begin
      Timer1.Enabled := false;
      XComm1.CloseDevice;
      SpeedButton1.Caption := 'Connect';
      Konek := False;
      ComboBoxPort.Enabled := True;
      ComboBoxBaut.Enabled := True;
      ComboBoxFc.Enabled := True;
      ProgressBarSinyal.Position := 0;
      ProgressBarBattery.Position := 0;

      ADODataSet1.Active:=False;
      ADODataSet2.Active:=False;
      ADOConnection1.Connected:=false;
    end
    else begin
      XComm1.BaudValue := StrToInt(ComboBoxBaut.Text);
      XComm1.DeviceName := ComboBoxPort.Text;
      XComm1.FlowControl := TFlowcontrol(ComboBoxFc.ItemIndex);
      If Not GoKonek Then begin
        sYangMoDiomongin:='Gagal membuka Port' +ComboBoxPort.Text;
        omongin(sYangMoDiomongin);
        ShowMessage('Gagal membuka port');
        end
        else
        SetTerkoneksi;
      end;
    end;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
  var s, c: string;
      p : integer;
  function getPhoneInfoIndex(info: string): integer;
    var i, r : integer;
  begin
    r := -1;
    for i := 0 to JumlahPhoneInfo - 1 do
    if PhoneInfo[i] = info then r := i;
    Result := r;
  end;
begin
  if not Konek then exit;
  if SedangBaca then exit;
```

```
      SedangBaca := True;
      List.Text := SendGetData('AT+CIND?'+#13, sOk);
      s := List.Strings[2];
      c := copy(s, pos(':', s) + 2, length(s));
      c := AnsiReplaceStr(c, ',', #13#10);
      Item.Text := c;
      if c = '' then exit;
      if Item.Count < 2 then exit;

      p := getPhoneInfoIndex('signal');
      if p <> - 1 then
         ProgressBarSinyal.Position := StrToInt(Item.Strings[p]);

      p := getPhoneInfoIndex('battchg');
      if p <> - 1 then
         ProgressBarBattery.Position := StrToInt(Item.Strings[p]);

      p := getPhoneInfoIndex('chargerconnected');
      if p <> - 1 then
      If Item.Strings[p] = '1' Then
       indCharged.Font.Color := clRed
       else
       indCharged.Font.Color:= clSkyBlue;

      p := getPhoneInfoIndex('message');
      if p <> - 1 then
         If Item.Strings[p] = '1' Then begin
           indSMSBaru.Font.Color := clRed;
           ComboBoxSumber.ItemIndex := 0;
           CekSMS;
           ComboBoxSumber.ItemIndex := 1;
           CekSMS;
           sndPlaySound('Sound/Ada SMS.wav',SND_FILENAME);
           SimpanSMS;
         end
         else
         begin
           indSMSBaru.Font.Color := clSkyBlue;
           indStatus.Caption:='Stand By';
           indStatus.Font.Color:=clSkyBlue;
         end;
      SedangBaca := False;
end;

procedure TForm1.SpeedButton4Click(Sender: TObject);
  var   s,c: string;
        i,j,l1,l2,p1,p2 : integer;
begin
  if not Konek then begin
    sYangMoDiomongin:='Belum terkoneksi dengan ponsel';
    omongin(sYangMoDiomongin);
    ShowMessage('Belum terkoneksi ke HP');
    exit;
  end;

  s := SendGetData('AT+CPBS=' + arSumber[ComboBoxPbook.itemindex]+#
13, sOK);
  if Pos(sOK, s) = 0 then begin
    sYangMoDiomongin:='Tidak dapat memilih media'+
ComboBoxPbook.Text;
    omongin(sYangMoDiomongin);
    ShowMessage('Tidak dapat memilih media'+ ComboBoxPbook.Text);
    exit;
  end;
  s := SendGetData('AT+CPBR=?'+#13, sOK);
  if Pos(sOK, s) = 0 then begin
```

```
    sYangMoDiomongin:='Tidak dapat membaca buku telepon';
    omongin(sYangMoDiomongin);
    ShowMessage('Tidak dapat membaca buku telepon');
    exit;
  end;

  p1 := pos('(', s) + 1;
  p2 := pos(')', s);
  c := copy(s, p1, p2 - p1);
  p1 := pos('-', c);
  if p1 = 0 Then exit;

  Timer1.Enabled := False;
  l1 := StrToInt(copy(c, 1, p1 - 1));
  l2 := StrToInt(copy(c, p1 + 1, length(c)));
  ProgressBar1.Min := l1;
  ProgressBar1.Max := l2;
  ProgressBar1.Visible := true;
  j := 0;
  for i := l1 to l2 do begin
    ProgressBar1.Position := i;
    s := SendGetData('AT+CPBR=' + IntToStr(i)+#13, sOK);

    p1 := pos(':', s) + 2;
    s := Copy(s, p1, length(s));
    List2.Text := AnsiReplaceStr(s,',', #13#10);
    If List2.Count > 3 then begin
       if copy(TrimAll(List2.Strings[1]),1,1)='0' then  begin
        ADODataSet2.Active:=False;
        ADOTable2.Active:=True;
        ADOTable2.Append;
        ADOTable2['nama']:=TrimAll(List2.Strings[3]);
        ADOTable2['notelp']:=TrimAll(List2.Strings[1]);
        ADOTable2['jenis']:=ComboBoxPbook.Text;
        ADOTable2.Post;
        ADOTable2.Active:=False;
        ADODataSet2.CommandText:='select distinct nama,notelp,jenis
from phonebook';
        ADODataSet2.Active:=True;
        Inc(j);
       end
       else
       if copy(TrimAll(List2.Strings[1]),1,1)='+' then  begin
        ADODataSet2.Active:=False;
        ADOTable2.Active:=True;
        ADOTable2.Append;
        ADOTable2['nama']:=TrimAll(List2.Strings[3]);
        ADOTable2['notelp']:='0'+copy(TrimAll(List2.Strings
[1]),4,20);
        ADOTable2['jenis']:=ComboBoxPbook.Text;
        ADOTable2.Post;
        ADOTable2.Active:=False;
        ADODataSet2.CommandText:='select distinct nama,notelp,jenis
from phonebook';
        ADODataSet2.Active:=True;
        Inc(j);
       end
       else
       begin
        ADODataSet2.Active:=False;
        ADOTable2.Active:=True;
        ADOTable2.Append;
        ADOTable2['nama']:=TrimAll(List2.Strings[3]);
        ADOTable2['notelp']:=TrimAll(List2.Strings[1]);
        ADOTable2['jenis']:=ComboBoxPbook.Text;
        ADOTable2.Post;
```

```
        ADOTable2.Active:=False;
        ADODataSet2.CommandText:='select distinct nama,notelp,jenis
from phonebook';
        ADODataSet2.Active:=True;
        Inc(j);
       end;
    end;
  end;
  ProgressBar1.Visible := false;
  sYangMoDiomongin:=IntToStr(j)+'ponbuk terbaca';
  omongin(sYangMoDiomongin);
  ShowMessage(IntToStr(j) + ' phonebook terbaca');
  Timer1.Enabled := True;
end;
//=======================Delete==========================
procedure TForm1.SpeedButton6Click(Sender: TObject);
begin
  ADODataSet1.Active:=False;
  with ADOQuery1 do begin
    Active:=False;
    Close;
    SQL.Clear;
    SQL.Add('DELETE FROM inbox');
    ExecSQL;
    sndPlaySound('Sound\recycle.wav',SND_FILENAME);
  end;
  ADODataSet1.Active:=true;
end;

procedure TForm1.SpeedButton7Click(Sender: TObject);
begin
  ADODataSet2.Active:=False;
  with ADOQuery1 do begin
    Active:=False;
    Close;
    SQL.Clear;
    SQL.Add('DELETE FROM Phonebook');
    ExecSQL;
    sndPlaySound('Sound\recycle.wav',SND_FILENAME);
  end;
  ADODataSet2.Active:=true;
end;
//=======================End==========================
procedure TForm1.DBGrid1CellClick(Column: TColumn);
begin
  Memo2.Text:=ADODataSet1['tanggal']+' '+ADODataSet1['notelp_nama']
+' '
  +ADODataSet1['isi']+' '+ADODataSet1['status'];
end;

procedure TForm1.TimerCheckStatusSMSTimer(Sender: TObject);
begin
  sndPlaySound('Sound/Ada SMS.wav',SND_FILENAME);
end;

procedure TForm1.Memo2Change(Sender: TObject);
  var p, s : integer;
begin
  p := ((length(memo2.Text) - 1) div 160) + 1;
  s := length(memo2.Text) mod 160;
  if (s = 0) and (length(memo2.Text) <> 0) then s := 160;
  LabelSisaSMS.Font.Color:=clHotLight;
  LabelSisaSMS.Caption := IntToStr(160 - s) + ' (' + IntToStr(p) + '
sms)';
end;
```

```
procedure TForm1.TimerStatusTTSTimer(Sender: TObject);
begin
  if AdaTukangOmong then
  begin
    LabelStatusTTS.Font.Color:=clHotLight;
    LabelStatusTTS.caption:= 'Ada TTS';
    sndPlaySound('Sound\chimes.wav',SND_FILENAME);
    TimerStatusTTS.Enabled:=false;
  end;
  if not AdaTukangOmong then
  begin
    LabelStatusTTS.Font.Color:=clRed;
    LabelStatusTTS.caption:= 'Tidak Ada TTS';
    sndPlaySound('Sound\wrong.wav',SND_FILENAME);
  end;
end;

procedure TForm1.SpeedButton5Click(Sender: TObject);
begin
  sYangMoDiomongin:='Aktifasi kiy';
  omongin(sYangMoDiomongin);
  Form4.ShowModal;
end;

procedure TForm1.CheckBox1Click(Sender: TObject);
begin
  if CheckBox1.Checked then
    begin
      Form1.WindowState := wsNormal;
      Labelmax.Font.Color:=clHotLight;
    end
  else
  Form1.WindowState := wsMaximized;
end;

procedure TForm1.FormDestroy(Sender: TObject);
begin
  with TRegistry.Create do try
    RootKey := HKEY_LOCAL_MACHINE;
    OpenKey('\SOFTWARE\Microsoft\Windows\CurrentVersion\Run', True);
    WriteString('PembacaSMS', Application.ExeName);
    CloseKey;
    finally
      Free;
  end;
end;
//
=========================OnKeyDown================================
procedure TForm1.ListViewPhoneKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = 13 then
  begin
   SpeedButton1.Click;
  end;
  if Key = vk_space then
  begin
   SpeedButton1.Click;
  end;
  if Key = Vk_Escape then
  begin
    Close;
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
```

```
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Down then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Up then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
end;
procedure TForm1.Memo2KeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = Vk_Escape then
  begin
    Close;
  end;
  if Key = 13 then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Down then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Up then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
end;

procedure TForm1.ComboBoxPortKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = 13 then
  begin
```

```
    SpeedButton1.Click;
   end;
   if Key = vk_space then
   begin
    SpeedButton1.Click;
   end;
   if Key = Vk_Escape then
   begin
     Close;
   end;
   if Key = Vk_Right then
   begin
     Omongin(Memo2.Lines.Text);
     Delay(30000);
     Memo2.Clear;
   end;
   if Key = Vk_Left then
   begin
     Omongin(Memo2.Lines.Text);
     Delay(30000);
     Memo2.Clear;
   end;
   if Key = Vk_Down then
   begin
     Omongin(Memo2.Lines.Text);
     Delay(30000);
     Memo2.Clear;
   end;
   if Key = Vk_Up then
   begin
     Omongin(Memo2.Lines.Text);
     Delay(30000);
     Memo2.Clear;
   end;
end;

procedure TForm1.DBGrid1KeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = 13 then
  begin
   SpeedButton1.Click;
  end;
  if Key = vk_space then
  begin
   SpeedButton1.Click;
  end;
  if Key = Vk_Escape then
  begin
    Close;
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Down then
  begin
    Omongin(Memo2.Lines.Text);
```

```
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Up then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
end;

procedure TForm1.DBGrid2KeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
if Key = 13 then
  begin
   SpeedButton1.Click;
  end;
  if Key = vk_space then
  begin
   SpeedButton1.Click;
  end;
  if Key = Vk_Escape then
  begin
    Close;
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Down then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Up then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
end;

procedure TForm1.ComboBoxPbookKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = 13 then
  begin
   SpeedButton1.Click;
  end;
  if Key = vk_space then
  begin
   SpeedButton1.Click;
  end;
  if Key = Vk_Escape then
  begin
    Close;
```

```
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Down then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Up then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
end;

procedure TForm1.ComboBoxSumberKeyDown(Sender: TObject; var Key:
Word;
  Shift: TShiftState);
begin
  if Key = 13 then
  begin
   SpeedButton1.Click;
  end;
  if Key = vk_space then
  begin
   SpeedButton1.Click;
  end;
  if Key = Vk_Escape then
  begin
    Close;
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Down then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Up then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
```

```
  end;
end;

procedure TForm1.CheckBox1KeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = 13 then
  begin
   SpeedButton1.Click;
  end;
  if Key = vk_space then
  begin
   SpeedButton1.Click;
  end;
  if Key = Vk_Escape then
  begin
    Close;
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Down then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Up then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
end;

procedure TForm1.ComboBoxBautKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = 13 then
  begin
   SpeedButton1.Click;
  end;
  if Key = vk_space then
  begin
   SpeedButton1.Click;
  end;
  if Key = Vk_Escape then
  begin
    Close;
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
```

```
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
    if Key = Vk_Down then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
    if Key = Vk_Up then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
end;

procedure TForm1.ComboBoxFcKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
    if Key = 13 then
    begin
     SpeedButton1.Click;
    end;
    if Key = vk_space then
    begin
     SpeedButton1.Click;
    end;
    if Key = Vk_Escape then
    begin
      Close;
    end;
    if Key = Vk_Right then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
    if Key = Vk_Left then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
    if Key = Vk_Down then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
    if Key = Vk_Up then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
end;

procedure TForm1.FormKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
    if Key = 13 then
    begin
     SpeedButton1.Click;
```

```
    end;
    if Key = vk_space then
    begin
     SpeedButton1.Click;
    end;
    if Key = Vk_Escape then
    begin
      Close;
    end;
    if Key = Vk_Right then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
    if Key = Vk_Left then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
    if Key = Vk_Down then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
    if Key = Vk_Up then
    begin
      Omongin(Memo2.Lines.Text);
      Delay(30000);
      Memo2.Clear;
    end;
end;

procedure TForm1.Memo1KeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = vk_space then
  begin
   SpeedButton1.Click;
  end;
  if Key = Vk_Escape then
  begin
    Close;
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Down then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Up then
  begin
```

```
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
end;

procedure TForm1.Edit1KeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = 13 then
  begin
   SpeedButton8.Click;
  end;
  if Key = vk_space then
  begin
   SpeedButton1.Click;
  end;
  if Key = Vk_Escape then
  begin
    Close;
  end;
  if Key = Vk_Right then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Left then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Down then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
  if Key = Vk_Up then
  begin
    Omongin(Memo2.Lines.Text);
    Delay(30000);
    Memo2.Clear;
  end;
end;
//============================End================================
procedure TForm1.SpeedButton8Click(Sender: TObject);
begin
ADODataSet2.Active:=False;
ADODataSet2.CommandText:='select distinct nama,notelp,jenis from
phonebook where Phonebook.nama LIKE'+ QuotedStr('%'+Edit1.Text+ '%');
ADODataSet2.Active:=True;
end;

procedure TForm1.SpeedButton9Click(Sender: TObject);
begin
ADODataSet2.Active:=False;
ADODataSet2.CommandText:='select distinct nama,notelp,jenis from
phonebook';
ADODataSet2.Active:=True;
end;

procedure TForm1.SpeedButton10Click(Sender: TObject);
begin
ADODataSet1.Active:=False;
```

```
ADODataSet1.CommandText:='select notelp_nama,tanggal,isi,status from
inbox order by tanggal Desc';
ADODataSet1.Active:=True;
end;

procedure TForm1.sBitBtn1Click(Sender: TObject);
begin
  sYangMoDiomongin:='Fom ebout';
  omongin(sYangMoDiomongin);
  Form2.ShowModal;
end;

procedure TForm1.sBitBtn2Click(Sender: TObject);
begin
  sYangMoDiomongin:='Fom petunjuk pengguna an';
  omongin(sYangMoDiomongin);
  Form3.ShowModal;
end;

end.
```

```
ADODataSet1.CommandText:='select notelp_nama,tanggal,isi,status from
inbox order by tanggal Desc';
ADODataSet1.Active:=True;
end;

procedure TForm1.sBitBtn1Click(Sender: TObject);
begin
  sYangMoDiomongin:='Fom ebout';
  omongin(sYangMoDiomongin);
  Form2.ShowModal;
end;

procedure TForm1.sBitBtn2Click(Sender: TObject);
begin
  sYangMoDiomongin:='Fom petunjuk pengguna an';
  omongin(sYangMoDiomongin);
  Form3.ShowModal;
end;

end.
```

```
unit Unit4;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, StdCtrls, Buttons, jpeg, ExtCtrls, sBitBtn;

type
  TForm4 = class(TForm)
    Edit1: TEdit;
    Label1: TLabel;
    sBitBtn1: TsBitBtn;
    procedure Edit1KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure sBitBtn1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form4: TForm4;

implementation

uses Unit1,pustakaUserSpeechInterface;

{$R *.dfm}
procedure TForm4.Edit1KeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key = 13 then
  begin
   sBitBtn1.Click;
  end;
end;

procedure TForm4.sBitBtn1Click(Sender: TObject);
begin
  if edit1.Text = '12345' then begin
    Form1.Labelkey.Caption:='5';
    Form1.Labelkey.Font.Color:=clHotLight;
    Form1.Labelsisa.Font.Color:=clHotLight;
    sYangMoDiomongin:='Masa pemakayan program hanya berlaku sampai
lima kali';
    Omongin(sYangMoDiomongin);
    ShowMessage('Masa pemakaian program hanya berlaku sampai 5
kali');
    edit1.Clear;
    close;
  end
  else
  if edit1.Text = '123' then begin
    Form1.Labelkey.Caption:='1';
    sYangMoDiomongin:='Selamat Mencoba anda hanya bisa menggunakan
program 1 kali saja';
    Omongin(sYangMoDiomongin);
    ShowMessage('Selamat mencoba');
    edit1.Clear;
    Close;
    end
  else
  if edit1.Text = '1205317' then begin
```

```
    Form1.Labelkey.Caption:='######';
    Form1.Labelkey.Font.Color:=clHotLight;
    Form1.Labelsisa.Font.Color:=clHotLight;
    sYangMoDiomongin:='Terimakasih telah menggunakan program ini';
    Omongin(sYangMoDiomongin);
    ShowMessage('Terimakasih telah menggunakan program ini');
    edit1.Clear;
    Close;
  end
  else begin
    sYangMoDiomongin:='Silahkan hubungi rozein terimakasih';
    Omongin(sYangMoDiomongin);
    ShowMessage('Silakan Hub Rozein di 085646429199. Terimaksih');
    edit1.Clear;
    Close;
  end;
  Form1.SimpanSetting;
end;

end.
```

```
unit gsm_sms;

interface

uses
  SysUtils, DateUtils, Dialogs, StrUtils;

type
  TSMS = class(TObject)
  private
    FIsSMSSumit: Boolean;
    FValidityLen: Integer;
    FSMSCLen: Integer;
    FSenderLen: Integer;
    FSenderPos: Integer;
    FPDU: String;
    FSMSDeliverStartPos: Integer;
    FMessage: WideString;
    FMessageRef: String;
    FAddress: String;
    FFlashSMS: Boolean;
    FRequestReply: Boolean;
    FDataCoding: Integer;
    FMessageLength: Integer;
    FIsUDH: Boolean;
    FUDHI: String;
    FStatusRequest: Boolean;
    FSizeOfPDU: integer;
    procedure SetPDU(const Value: String);
    function GetPDU: String;
    function ReverseOctets(Octets: String): String;
    function DecodeNumber(raw: String): String;
    function EncodeNumber(Number: String): String;
    function GetMessage: WideString;
    function GetAddress: String;
    function GetSMSC: String;
    function GetTimeStamp: TDateTime;
    function Get7bit(str: String): String;
    function Get8bit(str: String): String;
    function GetUCS2(str: String): WideString;
    function MakeCRLF(str: string): String;
    procedure Set_MessageRef(const Value: String);
  public
    dcs: Integer;
    function GetNewPDU(AMessageReference: String): String;
    property RequestReply: Boolean read FRequestReply write
FRequestReply;
    property PDU: String read GetPDU write SetPDU;
    property UDHI: String read FUDHI write FUDHI;
    property MessageReference: String read FMessageRef write
Set_MessageRef;
    property Text: WideString read GetMessage write FMessage;
    property Number: String read GetAddress write FAddress;
    property SMSC: String read GetSMSC;
    property FlashSMS: Boolean read FFlashSMS write FFlashSMS;
    property StatusRequest: Boolean read FStatusRequest write
FStatusRequest;
    property IsOutgoing: Boolean read FIsSMSSumit;
    property IsUDH: Boolean read FIsUDH;
    property TimeStamp: TDateTime read GetTimeStamp;
    property TPLength: integer read FSizeOfPDU;
  end;

const
  DoStrictUCScheck: boolean = True;
  ForceUCSusage: boolean = False;
```

```
function CheckCodingType(str: WideString): Integer;
function ConvertCharSet(inputStr: String; toGSM: Boolean): String;
overload; // from GSM mode only!
function ConvertCharSet(inputChr: Char; toGSM: Boolean = False):
Char; overload;

{ implementation found on interget, might solve chinese utf8 problems
?
function DecodeUTF8(const Value : string):string;
function EncodeUTF8(const Value : string):string;
}

implementation

uses Windows, Unit1;

{ UTF8 }

function DecodeUTF8(const Value : string):string;
var i, j : integer;
    N : integer;
    HugeChar : ULONG; //4 bytes
begin
  Result:='';
  i:=1;
  while i < Length(Value) do begin
    if byte(Value[i]) < $80 then begin
      Result:=Result+Value[i]; //no change required
      i:=i+1;
    end
    else begin
      //find out the number of bytes used for this character
      N:=0;
      for j:=1 to 8 do begin
      //start with the highest bit and cound the bumber
      //of "1" before "0"
        if (byte(Value[i]) and (1 shl (8-j))) = 0 then Break;
        inc(N);
      end;
      //ShowMessage('N:'+IntToStr(N));
      HugeChar:=byte(Value[i]) and ($FF shr (N+1));
      //ShowMessage('HugeChar:'+IntToStr(HugeChar));
      for j:=1 to N-1 do begin
        HugeChar:=(HugeChar shl 6) or byte(byte(Value[i+j]) and $3F);
      end;
      //ShowMessage('HugeChar:'+IntToStr(HugeChar));
      Result:=Result+char(HugeChar);
      i:=i+N;
    end;
  end;
end;

//only work on bytes 0..255
function EncodeUTF8(const Value : string):string;
var i : integer;
begin
  for i:=1 to Length(Value) do begin
    if byte(Value[i]) < $80 then begin
      Result:=Result+Value[i]; //no change required
    end
    else begin
      Result:=Result+char($C0{11000000} or (byte(Value[i]) shr 6))+
                     char($80{10000000} or (byte(Value[i]) and $3F
{111111}));
    end;
```

```
  end;
end;

{ TSMS }

function TSMS.DecodeNumber(raw: String): String;
var
  addrType: Integer;
begin
  try
    addrType := StrToInt('$' + copy(Raw, 1, 2));
    if ((addrType and $50) = $50) then begin
      Result := Get7bit(copy(Raw, 3, length(Raw) - 2));
    end
    else begin
      Result := ReverseOctets(copy(Raw, 3, length(Raw) - 2));
      if Result[length(Result)] = 'F' then Result := copy(Result, 1,
length(Result) - 1);
      if ((StrToInt('$' + copy(Raw, 1, 2)) and $70) shr 4) = 1 then
Result := '+' + result;
    end;
  except
    Result := '';
  end;
end;

function TSMS.EncodeNumber(Number: String): String;
begin
  Result := '81';

  if Number[1] = '+' then begin
    Result := '91'; // International Numner, ISDN/Telephone
(E.164/E.163)
    Number := copy(Number, 2, length(Number));
  end;

  Result := IntToHex(length(Number), 2) + Result;

  if length(Number) mod 2 > 0 then Number := Number + 'F';
  Result := Result + ReverseOctets(Number);
end;

function TSMS.Get7bit(str: String): String;
var
  i, j, x: Integer;
  leftover, octet: byte;
  c: string[2];
begin
  Result := '';
  x := 1;
  leftover := 0;
  j := Round(length(str) / 2) - 1;

  for i := 0 to j do begin
    try
      c := copy(str, (i*2)+1, 2);
      if not (Copy(c,1,1)[1] in ['0'..'9','A'..'F']) then
        break;
      if (Length(c) = 2) and not (Copy(c,2,1)[1] in
['0'..'9','A'..'F']) then
        Delete(c,2,1);
      octet := StrToInt('$' + c);
      Result := Result + chr(((octet and ($FF shr x)) shl (x - 1)) or
leftover);
      leftover := (octet and (not ($FF shr x))) shr (8 - x);
      x := x + 1;
```

```
    except
    end;

    if x = 8 then begin
      { do not add extra @ at the end of text, bug 849905 fixed }
      if (i <> j) or (leftover <> 0) then
        Result := Result + chr(leftover);
      x := 1;
      leftover := 0;
    end;
  end;

  Result := ConvertCharSet(Result, false);
end;

function TSMS.Get8bit(str: String): String;
var
  i: Integer;
  octet: Integer;
begin
  Result := '';

  for i := 0 to Round(length(str) / 2) - 1 do begin
    octet := StrToInt('$' + copy(str, (i*2)+1, 2));
    Result := Result + chr(octet);
  end;

  Result := ConvertCharSet(Result, false);
end;

function TSMS.GetUCS2(str: String): WideString;
var
  i: Integer;
  octet: Integer;
begin
  Result := '';

  while (length(str) mod 4) <> 0 do str := str + '0';

  for i := 0 to (length(str) div 4) - 1 do begin
    octet := StrToInt('$' + copy(str, (i*4)+1, 4));
    Result := Result + Widechar(octet);
  end;
end;

function TSMS.GetMessage: WideString;
var
  startpos: Integer;
  str, UDHnull: String;
  UDHIlength, i :Integer;
  function RemoveTail00(s: string): string;
  var
    i: integer;
  begin
    i := Length(s);
    if i >= 2 then begin
      if Copy(s,i-1,2) = '00' then
        Delete(s,i-1,2);
    end;
    Result := s;
  end;
begin
  try
    Result := '';
    UDHILength := 0;
```

```
    startpos := FSMSDeliverStartPos + FSenderLen + FValidityLen + 12;
    if not FIsSMSSumit then startpos := startpos + 12;

    if FIsUDH then begin
      UDHILength := StrToInt('$' + copy(FPDU, startpos + 2, 2));

      FUDHI := copy(FPDU, startpos + 2, UDHILength * 2 + 2);
      //Replace UDH with NULL chars
      for i:=0 to UDHILength do begin
         UDHnull := UDHnull + '00';
      end;
      Delete(FPDU,startpos + 2,UDHILength * 2 + 2);
      Insert(UDHNull,FPDU,startpos + 2);
      //FPDU := AnsiReplaceStr(FPDU, FUDHI, UDHNull);
    end;

    // TP-User-Data-Length. Length of message. The TP-DCS field
indicated 7-bit data, so the length here is the number
    // of septets. If the TP-DCS field were set to 8-bit data or
Unicode, the length would be the number of octets.
    FMessageLength := StrToInt('$' + copy(FPDU, startpos, 2));

    if FDataCoding = 0 then begin
       str := copy(FPDU, startpos + 2, length(FPDU)); // process the
rest of PDU data, will cut the message length later
       Result := Get7bit(str);
       // here FMessageLength contains number of septets (decoded
chars)
       if FIsUDH then
         Result := Copy(Result, ((UDHILength div 7) + UDHILength + 2)
+ 1, FMessageLength)
       else
         Result := Copy(Result, 1, FMessageLength);
    end
    else if FDataCoding = 1 then begin
       // here FMessageLength contains numbers of octets (encoded
bytes)
       str := copy(FPDU, startpos + 2, (FMessageLength)*2);
       Result := Get8bit(str);
       if FIsUDH then
         Result := Copy(Result, ((UDHILength div 7) + UDHILength + 2)
+ 1, Length(Result));
    end
    else if FDataCoding = 2 then begin
       // here FMessageLength contains numbers of octets (encoded
bytes)
       str := copy(FPDU, startpos + 2, (FMessageLength)*2);
       Result := GetUCS2(str);
       if FIsUDH then begin
         i := ((UDHILength + 1) mod 4) + 2;
         Result := Copy(Result, i, Length(Result));
         { TODO: unicode support instead of copy? }
         //Result := WideCharLenToString(@Result[i], Length(Result) -
i + 1);
       end;
    end
    else Result := '(Unsupported: Unknown coding scheme)';

    Result := MakeCRLF(Result);
  except
    Result := '(Decoding Error)';
  end;
end;

function TSMS.GetPDU: String;
var
```

```
  udhl: Integer;
  i, j, x, head: Integer;
  Octet: String;
  nextChr: Byte;

  pduAddr, pduMsgL, pduMsg: String;
  pduSMSC, pduFirst, pduMsgRef, pduPID, pduDCS, pduTPVP: String;
  AMessage: WideString;
begin
  AMessage := FMessage;
  udhl := 0;
  try
    // Convert Address (Destination No)
    pduAddr := EncodeNumber(FAddress);

    pduMsg := '';
    if (dcs = 0) or ((dcs = -1) and (CheckCodingType(AMessage) = 0))
then begin // 7-bit coding
      // Convert Message
      if FUDHI <> '' then begin
         udhl := StrToInt(Copy(FUDHI,1,2));
         udhl := (udhl div 7) + udhl + 2;
         for i:=0 to udhl - 1 do begin
            AMessage := '@' + AMessage;
         end;
      end;
      pduMsgL := IntToHex(length(AMessage), 2); // number of
septets!! IT IS NOT NUMBER OF OCTETS IN 7-bit encoding mode!!

      x := 0;
      j := length(AMessage);
      for i := 1 to j do begin
        if x < 7 then begin
          if i = j then
            nextChr := 0
          else
            nextChr := Ord(ConvertCharSet(Char(AMessage[i+1]),
True));

          Octet := IntToHex( ((nextChr and (not ($FF shl (x+1)))) shl
(7-x)) or (Ord(ConvertCharSet(Char(AMessage[i]), True)) shr x) , 2);
          pduMsg := pduMsg + Octet;

          x := x + 1;
        end
        else x := 0;
      end;

      pduDCS := '00';
    end
    else if (dcs = 1) or ((dcs = -1) and (CheckCodingType(AMessage) =
1)) then begin // 8-bit coding
      if FUDHI <> '' then begin
         udhl := StrToInt(Copy(FUDHI,1,2));
         udhl := (udhl div 7) + udhl + 2;
         for i:=0 to udhl - 1 do begin
            AMessage := '@' + AMessage;
         end;
      end;
      for i := 1 to length(AMessage) do begin
        pduMsg := pduMsg + IntToHex(ord(ConvertCharSet(Char(AMessage
[i]), True)), 2);
      end;
      pduMsgL := IntToHex(length(pduMsg) div 2,2); // number of
octets
```

```
      pduDCS := '04';
    end
    else begin // UCS2 Coding
      if FUDHI <> '' then begin
         udhl := StrToInt(Copy(FUDHI,1,2));
         udhl := ((udhl + 1) mod 4) + 3;
         for i:=0 to udhl - 1 do begin
            AMessage := '@' + AMessage;
         end;
         udhl := udhl*2 + 1; // adjust udhl according to UCS2 coding
      end;
      for i := 1 to length(AMessage) do begin
        pduMsg := pduMsg + IntToHex(ord(AMessage[i]), 4);
      end;
      pduMsgL := IntToHex(length(pduMsg) div 2,2); // number of
octets

      pduDCS := '08';
    end;
    if FFlashSMS then pduDCS[1] := '1'; // i.e. '1x' depending of
code scheme selected

    if FUDHI <> '' then begin
       pduMsg := Copy(pduMsg, (udhl-1) * 2 + 1, length(pduMsg));
    end;

    pduSMSC :=   '00'; // No SMSC Information
    pduFirst :=  '11'; // No Validity Field, SMS-Sumit, No UDH, No
Status Request
    head := StrToInt('$' + pduFirst);
    if FStatusRequest then head := head or $20; //ADD Yes
StatusRequest
    if FUDHI <> '' then head := head or $40;    //ADD Yes UDH
    if FRequestReply then head := head or $80;  //ADD Yes
ReplyRequest

    pduFirst := IntToHex(head, 2);
    pduMsgRef := '00'; // Let the phone set Msg Ref itself
    if FMessageRef <> '' then pduMsgRef := FMessageRef;
    pduPID := '00';

    pduTPVP := 'FF';

    Result := pduFirst + pduMsgRef + pduAddr + pduPID + pduDCS +
pduTPVP + pduMsgL;
    if FUDHI <> '' then begin
       Result := Result + FUDHI;
    end;
    Result := Result + pduMsg;

    FSizeOfPDU := Length(Result) div 2;

    Result := pduSMSC + Result;
  except
    raise Exception.Create('Error encoding PDU');
  end;
end;

function TSMS.GetAddress: String;
begin
  Result := DecodeNumber(copy(FPDU, FSenderPos, FSenderLen + 2));
end;

function TSMS.GetSMSC: String;
begin
  if FSMSCLen > 0 then Result := DecodeNumber(copy(FPDU, 3,
```

```
FSMSCLen))
  else Result := '';
end;

function TSMS.GetTimeStamp: TDateTime;
var
  str: String;
  year, month, day, hour, minute, second: Integer;
begin
  if FIsSMSSumit then Result := 0
  else begin
    str := ReverseOctets(copy(FPDU, FSMSDeliverStartPos + FSenderLen
+ 10, 12));

    Year :=   StrToInt(copy(str,  1, 2));
    Month :=  StrToInt(copy(str,  3, 2));
    Day :=    StrToInt(copy(str,  5, 2));
    Hour :=   StrToInt(copy(str,  7, 2));
    Minute := StrToInt(copy(str,  9, 2));
    Second := StrToInt(copy(str, 11, 2));

    Result := EncodeDateTime(Year+2000, Month, Day, Hour, Minute,
Second, 0);
  end;
end;

function TSMS.ReverseOctets(Octets: String): String;
var
  i: Integer;
  buffer: char;
begin
  i := 1;
  while i < length(Octets) do begin
    buffer := Octets[i];
    Octets[i] := Octets[i+1];
    Octets[i+1] := buffer;
    i := i + 2;
  end;

  result := Octets;
end;

procedure TSMS.SetPDU(const Value: String);
var
  PDUType, TPVPF: Byte;
  TPDCS: Integer;
  Offset: Integer;
begin
  {
  The following example shows how to send the message "hellohello" in
the PDU mode from a Nokia 6110.

  AT+CMGF=0     //Set PDU mode
  AT+CSMS=0     //Check if modem supports SMS commands
  AT+CMGS=23  //Send message, 23 octets (excluding the two initial
zeros)
  >0011000B916407281553F80000AA0AE8329BFD4697D9EC37<ctrl-z>

  There are 23 octets in this message (46 'characters'). The first
octet ("00") doesn't count, it is only an indicator of the length of
  the SMSC information supplied (0). The PDU string consists of the
following:

  Octet(s)            Description
  00                  Length of SMSC information. Here the length is
0, which means that the SMSC stored in the phone should be used.
```

```
                    Note: This octet is optional. On some phones
this octet should be omitted! (Using the SMSC stored in phone is thus
implicit)
  11                First octet of the SMS-SUBMIT message.
  00                TP-Message-Reference. The "00" value here lets
the phone set the message reference number itself.
  0B                Address-Length. Length of phone number (11)
  91                Type-of-Address. (91 indicates international
format of the phone number).
  6407281553F8      The phone number in semi octets (46708251358).
The length of the phone number is odd (11), therefore a trailing
                    F has been added, as if the phone number were
"46708251358F". Using the unknown format (i.e. the Type-of-Address
                    81 instead of 91) would yield the phone number
octet sequence 7080523185 (0708251358). Note that this has the
                    length 10 (A), which is even.
  00                TP-PID. Protocol identifier
  00                TP-DCS. Data coding scheme.This message is
coded according to the 7bit default alphabet. Having "04" instead of
                    "00" here, would indicate that the TP-User-Data
field of this message should be interpreted as 8bit rather than
                    7bit (used in e.g. smart messaging, OTA
provisioning etc).
  AA                TP-Validity-Period. "AA" means 4 days. Note:
This octet is optional, see bits 4 and 3 of the first octet
  0A                TP-User-Data-Length. Length of message. The TP-
DCS field indicated 7-bit data, so the length here is the number of
                    septets (10). If the TP-DCS field were set to
8-bit data or Unicode, the length would be the number of octets.
  E8329BFD4697D9EC37  TP-User-Data. These octets represent the
message "hellohello". How to do the transformation from 7bit septets
into
                    octets is shown here
  }
  FPDU := Value;

  // Check if PDU contain SMSC information
  try
    FSMSCLen := StrToInt('$' + copy(FPDU, 1, 2)) * 2; // length in
octets * 2 = number of chars
  except
    FSMSCLen := 0;
  end;
  FSizeOfPDU := (Length(FPDU) - FSMSCLen) div 2 - 1; // number of
chars - FSMSCLen's 2 chars

  FSMSDeliverStartPos := 3; // char number, first 2 represent
FSMSCLen octet
  if FSMSCLen > 0 then FSMSDeliverStartPos := FSMSDeliverStartPos +
FSMSCLen;

  // Check if SMS-Sumit or SMS-Deliver
  try
    {
    First octet of the SMS-DELIVER PDU
    The first octet of the SMS-DELIVER PDU has the following layout:

    Bit no  7        6        5        4        3        2       1
0
    Name    TP-RP    TP-UDHI  TP-SRI   (unused) (unused) TP-MMS  TP-
MTI  TP-MTI

    Name    Meaning
    TP-RP   Reply path. Parameter indicating that reply path exists.
    TP-UDHI User data header indicator. This bit is set to 1 if the
User Data field starts with a header
```

```
    TP-SRI  Status report indication. This bit is set to 1 if a
status report is going to be returned to the SME
    TP-MMS  More messages to send. This bit is set to 0 if there are
more messages to send
    TP-MTI  Message type indicator. Bits no 1 and 0 are both set to 0
to indicate that this PDU is an SMS-DELIVER
    }
    PDUType := StrToInt('$' + copy(FPDU, FSMSDeliverStartPos, 2));
  except
    PDUType := 0;
  end;
  FIsSMSSumit := (PDUType and 3) = 1;
  //Check there are Header Information
  FIsUDH := (PDUType and 64) = 64;
  // Get Validity Field Length
  FValidityLen := 0;
  Offset := 0;
  if FIsSMSSumit then begin
    TPVPF := (PDUType and $18) shr 3;

    case TPVPF of
      1: FValidityLen := 14;
      2: FValidityLen := 2;
      3: FValidityLen := 14;
    else FValidityLen := 0;
    end;

    Offset := 2;
  end;

  // Get Sender Field Length and Startpos
  FSenderPos := FSMSDeliverStartPos + Offset + 4;
  try
    FSenderLen := StrToInt('$' + copy(FPDU, FSenderPos - 2, 2)); //
count of sender's number digits
  except
    FSenderLen := 0;
  end;
  if (FSenderLen mod 2) > 0 then FSenderLen := FSenderLen + 1;

  FMessageRef := Copy(FPDU, FSMSDeliverStartPos + 2, 2);
  try
    {
    The Type-of-Address octet indicates the format of a phone number.
The most common value of this octet
    is 91 hex (10010001 bin), which indicates international format. A
phone number in international format
    looks like 46708251358 (where the country code is 46). In the
national (or unknown) format the same
    phone number would look like 0708251358. The international format
is the most generic, and it has to
    be accepted also when the message is destined to a recipient in
the same country as the MSC or as the SGSN.

    Using the unknown format (i.e. the Type-of-Address 81 instead of
91) would yield the phone number octet
    sequence 7080523185 (0708251358). Note that this has the length
10 (A), which is even.
    }
    TPDCS := StrToInt('$' + copy(FPDU, FSenderPos + FSenderLen + 4,
2));
  except
    TPDCS := 0;
  end;
  {
    Should check DCS for $00abxxzz, where
```

```
       a = compression flag
       b = message class meaning
      xx = message data coding
      zz = message class

    So we are going to check bits 2 and 3 only ($00001100 = $C)
  }
  FDataCoding := (TPDCS and $0C) shr 2;
end;

//*const
//  Table: array[0..255] of byte =
// ( // 0    1    2    3    4    5    6    7    8    9    A    B    C
D    E    F
//{0}   64, 163,  36, 165, 232, 233, 250, 236, 242, 199,  10, 216,
248,  13, 197, 229,
//{1}    0,  95,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
198, 230, 223, 202,
//{2}   32,  33,  34,  35, 164,  37,  38,  39,  40,  41,  42,  43,
44,  45,  46,  47,
//{3}   48,  49,  50,  51,  52,  53,  54,  55,  56,  57,  58,  59,
60,  61,  62,  63,
//{4}  161,  65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,
76,  77,  78,  79,
//{5}   80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90, 196,
214, 209, 220, 167,
//{6}  191,  97,  98,  99, 100, 101, 102, 103, 104, 105, 106, 107,
108, 109, 110, 111,
//{7}  112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 228,
246, 241, 252, 224,
//{8}    0,   0,   0,   0,   0,   0,   0,   0, 183,   0,   0,   0,
0,   0,   0,   0,
//{9}    0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
0,   0,   0,   0,
//{A}    0, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171,
172,   0, 174, 175,
//{B}  176, 177, 178, 179, 180, 181, 182,   0, 184, 185, 186, 187,
188, 189, 190, 191,
//{C}  192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203,
204, 205, 206, 207,
//{D}  208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219,
220, 221, 222, 223,
//{E}  224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235,
236, 237, 238, 239,
//{E}  240, 241, 242, 243, 244, 245, 246,   0,   0, 249, 250, 251,
252, 253, 254, 255
//  );

const
  Table: array[0..255] of byte =
  ( // 0    1    2    3    4    5    6    7    8    9    A    B    C
D    E    F
{0}   64, 163,  36, 165, 232, 233, 250, 236, 242, 199,  10, 216, 248,
13, 197, 229,
{1}  196,  95, 214, 195, 203, 217, 208, 216, 211, 200, 206,   0, 198,
230, 223, 202,
{2}   32,  33,  34,  35, 164,  37,  38,  39,  40,  41,  42,  43,  44,
45,  46,  47,
{3}   48,  49,  50,  51,  52,  53,  54,  55,  56,  57,  58,  59,  60,
61,  62,  63,
{4}  161,  65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,
77,  78,  79,
{5}   80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90, 196, 214,
209, 220, 167,
{6}  191,  97,  98,  99, 100, 101, 102, 103, 104, 105, 106, 107, 108,
109, 110, 111,
```

```
{7}  112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 228, 246,
241, 252, 224,
{8}    0,   0,   0,   0,   0,   0,   0,   0, 183,   0,   0,   0,   0,
0,   0,   0,
{9}    0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
0,   0,   0,
{A}    0, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172,
0, 174, 175,
{B}  176, 177, 178, 179, 180, 181, 182,   0, 184, 185, 186, 187, 188,
189, 190, 191,
{C}  192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204,
205, 206, 207,
{D}  208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220,
221, 222, 223,
{E}  224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236,
237, 238, 239,
{E}  240, 241, 242, 243, 244, 245, 246,   0,   0, 249, 250, 251, 252,
253, 254, 255
  );

var
  GEscaped: Boolean;

function ConvertCharSet(inputChr: Char; toGSM: Boolean = False):
Char;
var
  i: Integer;
  found: Boolean;
begin
  Result := inputChr;

  if toGSM then begin
    found := False;
    for i := 0 to 255 do begin
      if Table[i] = ord(inputChr) then begin
        Result := chr(i);
        found := True;
        break;
      end;
    end;

    if not found then begin
      case ord(inputChr) of
      12: Result := chr(10);
      91: Result := chr(60);
      92: Result := chr(47);
      93: Result := chr(62);
      94: Result := chr(20);
      123: Result := chr(40);
      124: Result := chr(64);
      125: Result := chr(41);
      126: Result := chr(61);
      164: Result := chr(101);
      else
        Result := chr(63);
      end;
    end;
  end
  else begin
    if ord(inputChr) = $1B then begin
      Result := chr(0);
      GEscaped := True;
    end
    else begin
      if GEscaped then begin
        GEscaped := false;
```

```
        case ord(inputChr) of
          10: Result := chr(12);
          20: Result := chr(94);
          40: Result := chr(123);
          41: Result := chr(125);
          47: Result := chr(92);
          60: Result := chr(91);
          61: Result := chr(126);
          62: Result := chr(93);
          64: Result := chr(124);
          101: Result := chr(164);
        else
          Result := chr(0);
        end;
      end
      else Result := chr(Table[Ord(inputChr)]);
    end;
  end;

end;

function ConvertCharSet(inputStr: String; toGSM: Boolean): String;
var
  i, v: Integer;
  escaped: Boolean;
begin
  Result := '';

  if toGSM then begin
    for i := 0 to length(inputStr) do
      Result := Result + ConvertCharSet(inputStr[i],toGSM);
  end
  else begin
    escaped := false;
    for i := 1 to length(inputStr) do begin
      v := ord(inputStr[i]);

      if escaped then begin
        escaped := false;
        case v of
          10: v := 12;
          20: v := 94;
          40: v := 123;
          41: v := 125;
          47: v := 92;
          60: v := 91;
          61: v := 126;
          62: v := 93;
          64: v := 124;
         101: v := 164;
        else
          v := 0;
        end;

        Result := Result + chr(v);
      end
      else begin
        if v <> $1B then Result := Result + chr(Table[v])
        else escaped := true;
      end;
    end;
  end;
end;
```

```
function TSMS.MakeCRLF(str: string): String;
var
  i: Integer;
  skipnext: boolean;
begin
  Result := '';
  skipnext := false;

  for i := 1 to length(str) do begin
    if skipnext then skipnext := false
    else begin
      // check if already CRLF paired
      if ((str[i] = #$0A) and (str[i+1] = #$0D)) or ((str[i] = #$0D)
and (str[i+1] = #$0A)) then begin
        Result := Result + #$0D + #$0A;
        skipnext := true;
      end
      else if ((str[i] = #$0A) or (str[i] = #$0D)) then begin
        Result := Result + #$0D + #$0A;
      end
      else begin
        Result := Result + str[i];
      end;
    end;
  end;
end;

function CheckCodingType(str: WideString): Integer;
var
  str8bit: AnsiString;
  i: Integer;
begin
  Result := 0;
  str8bit := str;
  if str8bit <> str then
    Result := 2
  else begin
    for i := 1 to length(str8bit) do begin
      { workaround for UCS-2 excoding of cyrilic (and other i18n)
chars }
      if ForceUCSusage or (DoStrictUCScheck and (str8bit[i] in
['À'..'ß','à'..'ÿ'])) then begin
        Result := 2;
        break;
      end;
      if (ord(ConvertCharSet(str8bit[i], True)) and $80) = $80 then
begin
        Result := 1;
        break;
      end;
    end;
  end;
end;

procedure TSMS.Set_MessageRef(const Value: String);
begin
  FMessageRef := Copy(Value,1,2);
  while Length(FMessageRef) < 2 do FMessageRef := '0' + FMessageRef;
end;

function TSMS.GetNewPDU(AMessageReference: String): String;
begin
  MessageReference := AMessageReference;

  FSMSCLen := StrToInt('$' + copy(FPDU, 1, 2)) * 2;
  FSizeOfPDU := (Length(FPDU) - FSMSCLen) div 2 - 1;
```

```
  FSMSDeliverStartPos := 3;
  if FSMSCLen > 0 then FSMSDeliverStartPos := FSMSDeliverStartPos +
FSMSCLen;

  FPDU[FSMSDeliverStartPos + 2] := FMessageRef[1];
  FPDU[FSMSDeliverStartPos + 3] := FMessageRef[2];

  Result := FPDU;
end;

end.
```

```
{=============================================================================
=========
  @Autor        : KIOSS Project, 2002, Luri Darmawan
                  http://www.kioss.com
  @Modul        : pustakaUserSpeechInterface.pas
  @Dibuat       : 2003-01-08 19:21:58
  @Deskripsi    : unit ini digunakan sebagai antarmuka antara
aplikasi
                  yang kamu buat dengan TukangOmong.
                  Detil cara penggunaan bisa kamu cari di situs KIOSS
Project
                  di http://www.kioss.com
  @Kopirait     : KIOSS Project
  @Versi        : 1.0.0
  @Open Issues  :
  @Histori      :

=============================================================================
========}
unit pustakaUserSpeechInterface;

interface
uses

  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls;

const
  WM_KIRIMTEKS                      = WM_USER + 2003;
  cs_KelasTukangOmong               = 'TfSpeech';
  cs_TitelTukangOmong               = 'KIOSS Speech Interface [aka]
TukangOmong';
  YA                                = TRUE;
  TIDAK                             = FALSE;

function  AdaTukangOmong():boolean;
function  Omongin( psYangDiomongin:string):word;

var
  hHandleTukangOmong                : HWND;

implementation
//
=============================================================================
======//
function  AdaTukangOmong():boolean;
{  Deskripsi : ngecek keberadaan AdaTukangOmong

-----------------------------------------------------------------------
-------}
begin
  result                            := TIDAK;
  hHandleTukangOmong                := FindWindow( pchar(
cs_KelasTukangOmong),
                                       pchar( cs_TitelTukangOmong));
  if hHandleTukangOmong <> 0 then
    result                          := YA;
end;
//
=============================================================================
======//
function  Omongin( psYangDiomongin:string):word;
{  Deskripsi : kirim teks yang mo diomongim ke TukangOmong

-----------------------------------------------------------------------
```

```
-------}
begin
  result                              := 255;
  // cari, TukangOmong-nya ada pa nggak?
  if AdaTukangOmong then
  begin
    // window pa bukan, jangan-jangan ada yng mo nge-bajak
    if isWindow( hHandleTukangomong) then
    begin
      // kirim ke TukangOmong, InsyaAllah diomongin...!!!
      PostMessage ( hHandleTukangOmong, WM_KIRIMTEKS,
                  Integer( GlobalAddAtom(pchar(
psYangDiomongin))),0);
    end;
    result                            := 0;
  end;
end;

end.
```