

# RANCANG BANGUN PEMROGRAMAN SISTEM KENDALI PC SERVER JARAK JAUH MENGUNAKAN SMS GATEWAY BERBASIS OPEN SOURCE

Krisnha Prasetyo, Spd<sup>1</sup>, Suyatno Budiharjo, ST<sup>2</sup>, Triady Widodo<sup>3</sup>

<sup>1,3</sup>Akademi Teknik Telekomunikasi Sandhy Putra Jakarta, <sup>2</sup>PTIK

## ABSTRAK

SMS telah menjadi suatu tren, bahkan gaya hidup baru tersendiri saat ini. Kepopuleran dan kemudahan penggunaan teknologi SMS, telah menggerakkan banyak pihak untuk memanfaatkan teknologi ini. Diantara implementasi teknologi SMS dalam kehidupan sehari-hari saat ini adalah *remote control system* (sistem kendali jarak jauh), yang berfungsi untuk menjalankan sebuah aplikasi / program lain jarak jauh via SMS, dengan menggunakan SMS gateway sebagai *interface* sistem operasi komputer dengan perangkat komunikasi. Salah satu aplikasi dari implementasi teknologi SMS tersebut yaitu *Remote control system* pada PC Server. Untuk menjamin efisiensi dan efektifitas pada implementasi tersebut, saat ini banyak digunakan perangkat lunak berupa sistem operasi dan aplikasi yang murah, *free* namun andal berbasis *open source*, yaitu GNU/Linux serta beberapa aplikasi di dalamnya, diantaranya *tool* SMS gateway, bahasa pemrograman *Gambas*, dan *service-service system*.

Metodologi penelitian pada proyek akhir ini meliputi studi literatur, perancangan dan realisasi, pengujian dan analisa perancangan guna mencapai suatu hasil yang diharapkan, yaitu bekerjanya suatu sistem *remote control* PC Server.

Hasil akhir dari proyek akhir ini yaitu dapat menghasilkan suatu sistem yang dapat memudahkan *admin* komputer. Hal yang dapat dilakukan seorang *admin* hanyalah dengan mengetikkan SMS dari *handphone*-nya untuk bisa mengendalikan PC server walaupun jauh dari tempatnya.

**Kata Kunci :** SMS, SMS Gateway, Remote Control, PC Server, Open Source

## ABSTRACT

SMS have to be a trend, and so a new lifestyle even now. The famous easily used to SMS technology, have to move much party to the benefit this technology. In the middle of SMS technology in the livelihood is a remote control system, have function to way application / other program far distance by SMS, with using SMS gateway as interface computer operating system by communication set. Anyone application from SMS technology implementation is remote control system of Server PC.

To guarantee the efficiency and effectivity on that implementation, now much to used software are operating system and cheap application, free but realiable on the base *open source*, is GNU/Linux and some application inside, like tool SMS gateway, Gambas language program, and service system.

The research methodology of this final project overwhelming literature study, design and realization, testing and planning analyze to reach an expected result, is workable remote control system Server PC.

The last result of this final project are expected to give a system which can make easier the computer user ( admin ). Thing that can be done by a computer user (admin) is just to tick from handphone to control server PC although far from place.

**Key Words :** SMS, SMS Gateway, Remote Control, PC Server, Open Source

## PENDAHULUAN

### 1.1 Latar Belakang

*Short Message Service* yang lebih dikenal dengan SMS, telah menjadi suatu tren, bahkan gaya hidup baru tersendiri saat ini. Di mana-mana, baik di tempat keramaian maupun di lingkungan tempat tinggal, dapat dijumpai orang yang tengah asyik memainkan jari pada telepon genggam (ponsel) mereka, untuk menuliskan pesan singkat atau yang lebih dikenal dengan istilah SMS itu tadi.

SMS sendiri adalah sebuah teknologi. Teknologi yang memungkinkan untuk menerima maupun mengirim pesan antar telepon bergerak (ponsel). Teknologi ini pertama kali diperkenalkan pada tahun 1992 di Eropa oleh ETSI (*European Telecommunications Standards Institute*), dan pada awalnya menjadi suatu standar untuk telepon *wireless* berbasis GSM (*Global System for Mobile Communications*). Namun teknologi lain seperti CDMA dan TDMA pun memasukkan SMS ini sebagai fitur standar mereka.

Kepopuleran dan kemudahan penggunaan teknologi SMS, telah menggerakkan banyak pihak untuk memanfaatkan teknologi ini. Salah satu implementasi atau aplikasi teknologi SMS dalam kehidupan sehari-hari saat ini adalah *remote control system* (sistem kendali jarak jauh). Aplikasi ini berbeda dengan aplikasi-aplikasi lainnya seperti *polling / voting*, *reminder*, *warning*, *monitoring*, *autoresponder* dan sebagainya. Bila aplikasi tersebut dapat mengirimkan SMS, sedangkan *remote control system* berfungsi untuk menjalankan sebuah aplikasi / program lain jarak jauh via SMS, dengan menggunakan SMS gateway sebagai *interface* sistem operasi komputer dengan perangkat komunikasi.

Dalam sebuah kantor biasanya PC tidak berdiri sendiri, namun saling berhubungan dengan LAN ( *Local Area Network* ), yang dikontrol oleh PC server. PC server berfungsi mengendalikan *service-service* sistem seperti *server web*, *database server*, *proxy server*, *file sharing server* dan sebagainya. Misalnya terkadang *admin* lupa untuk mematikan, menghidupkan atau *restart* salah satu dari *service* sistem di atas pada PC servernya. Selain itu, apabila ia juga lupa untuk mematikan atau *restart* PC servernya. Mereka baru tersadar apabila telah meninggalkan PC server jauh dari tempatnya.

Alasan itulah yang membuat penulis berusaha untuk merancang suatu program *remote control system* pada PC Server yang dapat memecahkan masalah tersebut dengan menggunakan SMS gateway berbasis *Open Source* (perangkat lunak berupa sistem operasi dan aplikasi yang murah, *free* namun andal, dengan tujuan efisiensi dan efektifitas).

### 1.2 Tujuan Penulisan

Adapun tujuan penulisan dari proyek akhir ini adalah :

1. Mempelajari dan merealisasikan perancangan program sistem kendali PC server jarak jauh menggunakan SMS gateway berbasis *open source*.
2. Memperkenalkan sistem operasi dan aplikasi yang murah, *free* namun andal (GNU/LINUX- Knoppix 4.0.2)
3. Mempelajari arsitektur dan penyusunan program pada Gambas 1.9.20.
4. Mempelajari konsep *client/server*.
5. Mempelajari dan menjalankan aplikasi SMS gateway
6. Mempelajari arsitektur dan menjalankan *service-service* sistem dan *command line tool*.

### 1.3 Rumusan Masalah

Dengan memperhatikan identifikasi masalah di atas, maka penelitian akan dilakukan untuk menjawab permasalahan-permasalahan dibawah ini :

- Bagaimana merancang dan membuat sistem kendali PC server dengan menggunakan SMS gateway berbasis *open source*.
- Apakah rancangan sistem kendali PC server menggunakan SMS gateway berbasis *open source* dapat bekerja dengan baik.

### 1.4 Pembatasan Masalah

Ruang lingkup permasalahan dalam laporan proyek akhir ini hanya terbatas pada masalah-masalah sebagai berikut :

1. Perancangan arsitektur SMS gateway .
2. Pembuatan program sistem kendali PC server menggunakan program Gambas 1.9.20.
3. Tidak membahas instalasi LAN, Knoppix 4.0.2, Gambas 1.9.20, Apache, Samba. Hanya membahas teori dasar secara umum.
4. Tidak membahas secara detail konfigurasi *serial port*, hanya membahas setting *serial port* untuk operasi SMS gateway.
5. Hanya membuat perintah untuk mematikan, menghidupkan, dan *restart* (on, off, & restart) *server web*, *server file sharing* , serta *restart / turn off* PC server.

### 1.5 Metodologi Penelitian

Dalam pelaksanaan proyek akhir ini, penulis melakukan beberapa metode penelitian untuk merealisasikan proyek akhir ini, yaitu :

#### 1. Studi Literatur

Metode ini dilakukan dengan melakukan studi literatur di perpustakaan kampus atau di perpustakaan lain yang berhubungan dengan permasalahan yang akan dibahas, dan

membaca buku referensi serta mencari data di situs internet yang dapat mendukung realisasi proyek akhir ini.

## 2. Perancangan

Pada tahap ini penulis akan mencoba merancang program sistem kendali berdasarkan spesifikasi-spesifikasi yang diinginkan.

## 3. Pengujian

Untuk mengetahui apakah perancangan sesuai dengan spesifikasi yang diinginkan maka dilakukan pengujian. Pengujian dilakukan pada saat rancangan telah selesai.

## 4. Analisa dan Evaluasi

Dari kesalahan-kesalahan yang diketahui pada tahap pengujian, maka perlu dilakukan penganalisaan. Hal ini dimaksudkan agar pada tahap selanjutnya dapat dilakukan perancangan yang lebih memuaskan. Evaluasi juga dibutuhkan untuk mengetahui seberapa bagus sistem yang dirancang dan faktor apa saja yang dapat mempengaruhi kegagalan pada sistem tersebut.

### 1.6 Sistematika Penulisan

Secara umum sistematika penulisan proyek akhir ini terdiri dari bab-bab dengan metode penyampaian sebagai berikut :

#### BAB I PENDAHULUAN

Pada bab ini dikemukakan latar belakang masalah, maksud dan tujuan, rumusan masalah, pembatasan masalah, metodologi penelitian, sistematika penulisan.

#### BAB II SMS GATEWAY DAN SISTEM KENDALI PC SERVER BERBASIS OPEN SOURCE

Pada bab ini dibahas kerangka teori yang merupakan teori dasar yang mendukung proyek akhir ini dibuat.

#### BAB III PERANCANGAN PROGRAM SISTEM KENDALI PC SERVER BERBASIS OPEN SOURCE

Pada bab ini digambarkan subjek penelitian, perancangan program, dan cara kerja program secara keseluruhan.

#### BAB IV PENGUJIAN DAN ANALISA SISTEM KENDALI PC SERVER BERBASIS OPEN SOURCE

Pada bab ini pembatasan dilakukan untuk menguji dan menganalisa hasil perancangan serta implementasi dari program.

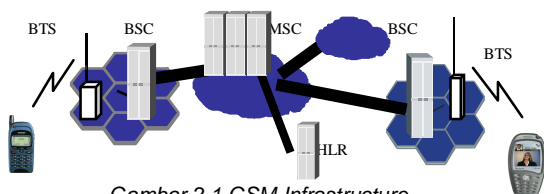
#### BAB V KESIMPULAN DAN SARAN

Pada bab ini dikemukakan kesimpulan dan saran-saran yang konstruktif untuk kesempurnaan proyek akhir.

#### SMS GATEWAY DAN SISTEM KENDALI PC SERVER BERBASIS OPEN SOURCE

##### 2.1 SMS (Short Message Service)

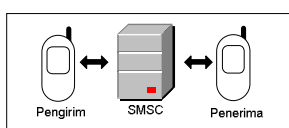
Short Message Service (SMS) merupakan salah satu fitur dari GSM yang dikembangkan dan di standarisasi oleh ETSI (European Telecommunications Standards Institute). Pada saat kita mengirim pesan SMS dari *handphone*, maka pesan SMS tersebut tidak langsung dikirim ke *handphone* tujuan, akan tetapi terlebih dahulu dikirim ke SMS Center (SMSC), yang terdapat di dalam MSC melalui sistem jaringan GSM dengan prinsip *Store and forward*, setelah itu baru dikirimkan ke *handphone* yang dituju.



Gambar 2.1 GSM Infrastructure

##### 2.1.1 Cara Kerja SMS

Melalui keberadaan SMSC, kita dapat mengetahui status dan SMS yang dikirim, apakah telah sampai atau gagal diterima oleh *handphone* tujuan. Apabila *handphone* tujuan dalam keadaan aktif dan menerima SMS yang dikirim, *handphone* penerima akan mengirim kembali pesan konfirmasi ke SMSC yang menyatakan bahwa SMS telah diterima. Kemudian SMSC mengirimkan status tersebut kepada pengirim. Proses pengiriman SMS dapat dilihat pada gambar dibawah ini.



Gambar 2.2 Cara Kerja SMS

##### 2.1.2 Protocol Data Unit (PDU)

Dalam pengiriman dan penerimaan pesan SMS terdapat 2 mode, yaitu mode teks dan mode *Protocol Data Unit* (PDU). Mode teks adalah format pesan dalam bentuk teks asli yang dituliskan pada saat akan mengirim pesan. Sesungguhnya mode teks ini adalah hasil pengkodean dari mode PDU. Sedangkan mode PDU adalah format pesan dalam bentuk oktet heksadesimal dan oktet semidesimal dengan panjang pesan mencapai 160 karakter (7 bit) atau 140 karakter (8 bit). Di Indonesia, tidak semua operator GSM maupun terminal mendukung mode teks, sehingga mode yang digunakan adalah mode PDU. Pada pengiriman pesan terdapat dua jenis mobile, yaitu *Mobile Terminated* (*handphone* Penerima) dan *Mobile Originated* (*handphone* Pengirim).

##### 2.1.2.1 SMS PDU Pengirim (*Mobile Originated*)

SMS PDU Pengirim adalah pesan yang dikirim dari *handphone* ke *terminal* melalui SMSC. Pesan yang akan dikirimkan oleh *handphone* masih dalam bentuk teks, sedangkan dalam pengiriman ke SMSC harus dalam bentuk PDU. Untuk itu sebelum dikirim, *handphone* akan melakukan perubahan dari format teks menjadi format PDU, proses ini sering disebut proses *encodec*. Adapun skema dan format PDU Pengirim telah diatur dan ditetapkan oleh ETSI sebagai berikut.

Tabel 2.1 Skema Format SMS PDU Pengirim

SCA	PDU Type	MR	DA	PID	DCS	VP	UDL	UD
-----	----------	----	----	-----	-----	----	-----	----

Misalnya kita mengirim pesan SMS ke nomor 628176609297 dengan isi pesan "ABC" dengan batas waktu pengiriman (waktu penyimpanan pesan di SMSC, jika nomor tujuan tidak dapat menerima pesan) 5 hari. Maka format PDU adalah :

**0011000C9126186706297900000341E110**

Berikut akan dijabarkan beberapa hal tentang Skema Format SMS PDU Pengirim, yaitu :

##### 1) Service Centre Address

SCA adalah informasi dan alamat (nomor) SMSC. SCA memiliki tiga komponen utama, yaitu *length*, *type of number*, dan *service centre number*. Dalam pengiriman pesan SMS, nomor SMSC tidak dicantumkan.

Tabel 2.2 Service Centre Address Pengirim

OKTET	KETERANGAN	HASIL
<i>Length</i>	Panjang informasi SMSC dalam oktet	00
<i>Type of number</i>	Format nomor dan SMSC 81 hexa = format lokal 91 hexa = format internasional	<none>
<i>Service Centre Number</i>	Nomor SMSC dan operator pengirim. Jika panjangnya ganjil maka pada karakter terakhir ditambahkan OF hexa	<none>

##### 2) PDU Type

Nilai default dan PDU Type untuk SMS pengirim adalah 11 hexa, yang memiliki arti bahwa 11 hexa = 00010001.

Tabel 2.3 PDU Type Pengirim

BIT NO	7	6	5	4	3	2	1	0
Name	RP	UDHI	SRR	VPF	VPF	RD	MTI	MTI
Nilai	0	0	0	1	0	0	0	1

Keterangan :

**RP** : *Reply Path*. Parameter yang menunjukkan bahwa alur jawaban ada.

**UDHI** : *User Data Header Indicator*. Bit ini bernilai 1 jika data pengirim dimulai dengan suatu judul/tema.

**SRR** : *Status Report Request*. Bit ini bernilai 1 jika laporan status pengiriman diminta.

**VPF** : *Validity Report Format*. Format dan batas waktu pengiriman jika pesan gagal diterima.

00 → Jika pesan tidak disimpan di SMSC

10 → Format relatif (satu oktet).

01 → Format enhanced (tujuh oktet).

11 → Format absolut (tujuh oktet).

**RD** : *Reject Duplicate*. Parameter yang menandakan ya atau tidaknya *Service Center* akan menerima suatu pengiriman pesan SMS untuk suatu pesan yang masih disimpan dalam *Service Center* tersebut. Ia mempunyai MR dan DA yang sama sebagai pesan dikirimkan dan OA yang sama.

**MTI** : *Message Type Indicator*. Bit bernilai 0 untuk menunjukkan bahwa PDU ini adalah suatu SMS-DELIVER.

##### 3) Message Reference (MR)

*Message Reference* adalah acuan dan pengaturan pesan SMS. Untuk membiarkan pengaturan pesan SMS dilakukan sendiri

oleh *handphone* tujuan, maka nilai yang diberikan adalah "00". Jadi pada *Message Reference* hasilnya adalah 00.

#### 4) Destination Address (DA)

DA adalah alamat (nomor tujuan, yang terdiri atas panjangnya nomor tujuan (*Length*), format dan nomor tujuan (*Type Number*), dan nomor tujuan (dan nomor tujuan (*Destination Number*)).

Tabel 2.4 Destination Address Pengirim

#### 5) Protocol Identifier (PID)

*Protocol Identifier* adalah tipe atau format dari cara pengiriman pesan. Misalnya tipe *Standart Text*, *Fax*, *E-mail*, *telex* dan lain-lainnya. Nilai *default* dari PID adalah 00 = "*Standart Text*". Pesan SMS yang akan dikirim menggunakan format teks standar, jadi pada *Protocol Identifier* hasilnya adalah 00.

#### 6) Data Coding Scheme (DCS)

*Data Coding Scheme* adalah rencana dan pengkodean atau *Blinking SMS*. Pesan SMS yang dikirim berupa teks standar, jadi pada *Data Coding Scheme* hasilnya adalah 00.

#### 7) User Data Length (UDL)

*User Data Length* adalah penjangnya pesan SMS yang akan dikirim dalam bentuk teks standar. Pada contoh ini pesan SMS yang dikirim adalah "ABC", yang memiliki 03 karakter (03 h). Jadi pada *User Data Length* hasilnya adalah 03.

#### 8) User Data (UD)

*User Data* adalah isi pesan yang akan dikirim dalam format heksadesimal. Pada contoh ini, isi pesan SMS yang dikirim adalah "ABC"=41E110. Pengkodean dari nilai teks standar menjadi heksadesimal dilakukan dengan bantuan Tabel ASCII.

#### 2.1.2.2 SMS PDU Penerima (Mobile Terminated)

SMS PDU Penerima adalah pesan dalam format PDU yang diterima *terminal* yang dikirim dari *handphone* melalui SMSC. Setelah itu *terminal* yang menerima pesan akan melakukan pengkodean menjadi teks, proses ini sering disebut proses *decodec*. Cara pengkodean format PDU sudah diatur dan distandarkan oleh ETSI. Format PDU dan SMS Penerima adalah :

Tabel 2.5 Skema Format PDU Penerima

SCA	PDU Type	OA	PID	DCS	SCTS	UDL	UD
-----	----------	----	-----	-----	------	-----	----

Contoh kita menerima pesan dari 628176609297 dengan isi pesan SMS adalah "hallo" pada tanggal 12 Juli 1985 pukul 07:24:02

OKTET	KETERANGAN	HASIL
<i>Length</i>	Panjang informasi SMSC dalam oktet	07
<i>Type of number</i>	Format nomor dan SMSC 81 hexa = format lokal 91 hexa = format internasional	91
<i>Service Center Number</i>	Nomor SMSC dan operator pengirim. Jika panjangnya ganjil maka pada karakter terakhir ditambahkan OF hexa Contoh : XL = 62818445009 (PDU = 2618485400F9)	2618485400F9

wib, maka format PDU adalah :

07912618485400F9040C912618670629790000

Penjelasan masing – masing format adalah sebagai berikut :

#### 1) Service Center Address (SCA)

SCA adalah alamat (nomor) dari SMSC. SCA memiliki tiga komponen utama, yaitu *Length*, *Type of number*, dan *Service Center Number*. Pada contoh di atas nilai dan SCA adalah **07912618485400F9**.

Tabel 2.6 Service Center Address Penerima

#### 2) PDU Type

Nilai *default* dan PDU Type untuk SMS-Deliver adalah **04** hexa, yang memiliki arti 04 hexa = 00000100.

Tabel 2.7 PDU Type Penerima

BIT NO	7	6	5	4	3	2	1	0
Name	RP	UDHI	SRI	<nn>	<nn>	MMS	MTI	MTI
Nilai	0	0	0	0	0	1	0	0

Keterangan :

**RP** : *Reply Path*. Parameter yang menunjukkan bahwa alur jawaban ada.

**UDHI** : *User Data Header Indicator*. Bit ini bernilai 1 jika data pengirim dimulai dengan suatu judul/tema.

**SRI** : *Status Report Indication*. Bit ini bernilai 1 jika suatu status laporan akan dikembalikan ke SMSC.

**MMS** : *More Message to Send*. Bit ini bernilai 0 jika ada pesan lebih yang akan dikirim.

**MTI** : *Message Type Indicator*. Bit ini bernilai 0 untuk menunjukkan bahwa PDU ini adalah suatu SMS Deliver.

#### 3) Originator Address (OA)

OA adalah alamat (nomor) dari pengirim, yang terdiri atas panjangnya nomor pengirim (*Length*), format nomor pengirim (*Type*

OKTET	KETERANGAN	HASIL
<i>Length</i>	12	0C
<i>Type of Number</i>	Format Internasional	91
<i>Destination number</i>	628176609297	261867062979

*Number*), dan nomor pengirim (*Originator Number*). Nilai OA pada contoh di atas adalah **0C91261867062979**.

Tabel 2.8 Originator Address Penerima

OKTET	KETERANGAN	HASIL
<i>Length</i>	Panjang nomor Pengirim	0C
<i>Type of number</i>	Format nomor dan SMSC 81 hexa = format lokal 91 hexa = format internasional	91
<i>Originator Number</i>	Nomor pengirim dan operator pengirim. Jika panjangnya ganjil, maka pada karakter terakhir ditambahkan OF hexa	261867062979

#### 4) Protocol Identifier (PID)

*Protocol Identifier* adalah tipe atau format dan cara pengiriman pesan, yang biasanya diatur dari *handphone* pengirim. Misalnya tipe *Standart Text*, *Fax*, *E-mail*, *Telex* dan lain-lainnya. Nilai *default* dan PID adalah **00** = "*Standart Text*". Untuk contoh di atas nilai dan PID adalah 00, sehingga pesan yang diterima berupa teks standar.

#### 5) Data Coding Scheme (DCS)

*Data Coding Scheme* adalah rencana dan pengkodean data untuk menentukan kelas dan pesan, apakah berupa SMS teks standar, *Flash SMS*, atau *Blinking SMS*. Pada contoh di atas DCS adalah **00** yang berarti bahwa pesan yang diterima merupakan pesan teks standar.

#### 6) Service Center Time Stamp (SCTS)

*Service Center Time Stamp* adalah waktu dan penerimaan pesan oleh SMSC penerima. SCTS terdiri atas tahun, bulan, tanggal, jam, menit, dan detik, serta zona waktu. Nilai SCTS pada contoh di atas adalah **58702170422082**.

Tabel 2.9 Service Center Time Stamp Penerima

NAMA	NILAI	HASIL
<i>Year</i>	85 (1985)	58
<i>Month</i>	07 (Juli)	70
<i>Date</i>	12	21
<i>Hour</i>	07	70
<i>Minute</i>	24	42
<i>Second</i>	02	20
<i>Time Zone</i>	28, dimana 1 unit 15 menit. Jadi (15x28)/60 – 7 jam. Sehingga menjadi GMT +07:00 – WIB.	82

#### 7) User Data Length (UDL)

*User Data Length* adalah panjang dari pesan yang diterima dalam bentuk teks standar. Pada contoh nilai dari UDL adalah **05**, yang berarti pesan yang diterima adalah sebanyak 5 karakter = "hallo".

#### 8) User Data (UD)

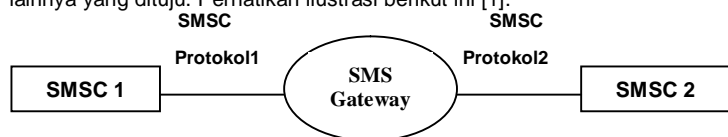
*User Data* adalah pesan yang diterima dalam format heksadesimal. Pada contoh di atas nilainya adalah **C8309BFD06**.

#### 2.2 SMS Gateway

Istilah *gateway*, bila dilihat pada kamus Inggris-Indonesia diartikan sebagai pintu gerbang. Namun pada dunia komputer, *gateway* dapat berarti juga sebagai jembatan penghubung antar satu sistem dengan sistem yang lain yang berbeda, sehingga dapat terjadi pertukaran data antar sistem tersebut. Dengan demikian, SMS *gateway* dapat diartikan sebagai suatu penghubung untuk lalu lintas data-data SMS, baik yang dikirimkan maupun yang diterima [1].

Pada awalnya, SMS *gateway* dibutuhkan untuk menjembatani antar SMSC. Hal ini dikarenakan SMSC yang dibangun oleh perusahaan yang berbeda memiliki protokol komunikasi sendiri, dan protokol-protokol itu sendiri bersifat pribadi. Sebagai contoh, Nokia

memiliki protokol SMSC yang disebut CIMD, sedangkan CMG memiliki protokol yang disebut dengan EMI. SMS gateway ini kemudian ditempatkan di antara kedua SMSC yang berbeda tersebut, yang berfungsi sebagai relay bagi keduanya, yang kemudian akan menerjemahkan data dari protokol SMSC satu ke protokol SMSC lainnya yang dituju. Perhatikan ilustrasi berikut ini [1].



Gambar 2.3 Ilustrasi SMS gateway

Namun seiring perkembangan teknologi komputer, baik dari sisi hardware maupun software, dan perkembangan teknologi komunikasi, SMS gateway tidak lagi dimaksudkan sebagaimana ilustrasi di atas. Dewasa ini, masyarakat lebih mengartikan SMS gateway sebagai suatu jembatan komunikasi yang menghubungkan perangkat komunikasi (dalam hal ini ponsel) dengan perangkat komputer mereka, yang menjadikan aktifitas SMS menjadi lebih mudah dan menyenangkan. Pengertian SMS gateway kemudian lebih mengarah pada suatu program yang mengkomunikasikan antara sistem operasi komputer, dengan perangkat komunikasi yang terpasang untuk mengirim atau menerima SMS. Salah satu komunikasi yang terjadi dapat dilakukan dengan mengirimkan perintah AT pada perangkat komunikasi tersebut, kemudian hasil operasinya dikirimkan kembali ke komputer. Dibutuhkan suatu *interface* baik dalam bentuk aplikasi maupun halaman *web* untuk dapat membaca SMS yang masuk, atau mengirim SMS tersebut [1].



Gambar 2.4 Aplikasi SMS gateway

### 2.2.1 Perangkat Komunikasi

Perangkat komunikasi di sini adalah perangkat yang dapat digunakan untuk mengirimkan atau menerima SMS. Perangkat-perangkat tersebut dapat berupa :

1. Telepon seluler (Ponsel)
2. GSM Modem
3. GPRS Modem

### 2.2.2 Media Koneksi

Media koneksi di sini adalah media yang dapat digunakan untuk mengirimkan data dari perangkat komunikasi ke komputer atau sebaliknya. Pada ponsel, istilah yang sering digunakan adalah *connectivity* atau konektivitas. Media-media tersebut di antaranya adalah :

1. Kabel Data
2. Bluetooth
3. IrDa

### 2.2.3 AT Command

AT Command adalah perintah – perintah yang digunakan dalam komunikasi dengan *Serial Port*. Dengan AT Command kita dapat mengetahui *vendor* dan *handphone* yang digunakan, kekuatan sinyal, membaca pesan yang ada pada *SIM Card*, mengirim pesan, mendeteksi pesan SMS baru yang masuk secara otomatis, menghapus pesan pada *SIM Card*, dan masih banyak lagi. Adapun perintah yang biasa digunakan yang berhubungan dengan SMS adalah sebagai berikut:

Tabel 2.10 AT Command Untuk SMS

AT Command	Keterangan
AT+CNMI	Mendeteksi pesan SMS baru yang masuk secara otomatis
AT+CMGL	Membuka daftar SMS yang ada pada SIM Card
AT+CMGS	Mengirim pesan SMS
AT+CMGR	Membaca pesan SMS
AT+CMGD	Menghapus pesan SMS

### 2.2.4 Komunikasi

Saat ini, proses komunikasi antar komputer dan *handphone* dilakukan melalui serial port USB dengan menggunakan standar serial UART (*Universal Asynchronous Receive Transmitter*)

#### 2.2.4.1 Setting

Software dapat mengontrol *setting* dari koneksi serial, biasanya seperti *setting baudrate/speed*, *parity*, *stopbits*, dan *flow control*.

#### 1. Baudrate

*Baudrate* adalah kecepatan *bit* dari satu *device* ke *device* yang lain dalam satu detik (*bit/s*). Biasanya kecepatan *bit* per detik untuk *synchronous start/stop communication* adalah 300, 1200, 2400, 9600, 19200 *bit/2*, dll.

#### 2. Parity

*Parity* adalah metode mendeteksi kesalahan/error pada transmisi. Tipe dari *setting parity* adalah :

*None* : Tidak adanya pengecekan *error* pada transmisi.

*Odd* : Adanya pengecekan *parity* dengan menambahkan *bit* per karakter per data unit (biasanya 7 atau 8 *bits*). *Bit Parity* akan di-set 0, jika total *bit* yang diterima tidak sama maka terjadi *error* atau *corrupt*.

*Even* : Adanya pengecekan *parity* dengan menambahkan 1 *bit* per karakter per data unit (biasanya 7 atau 8 *bits*), *bit parity* akan di-set 1, jika total *bit* yang diterima tidak sama maka terjadi *error* atau *corrupt*.

#### 3. DataBits

Menspesifikasikan banyaknya *bits* dalam satu kali pengiriman, biasanya menggunakan 5 sampai 8 *bits*.

#### 4. StopBits

Menspesifikasikan banyaknya *bits* yang mengindikasikan akhir dari 1 *byte*. *StopBits* biasanya menggunakan 1 atau 2 *bits*.

#### 5. Flow Control.

Merupakan pengontrolan *flow* data yang masuk atau keluar. *Serial Port* mungkin menggunakan sinyal pada *interface* untuk mem-pause dan menjalankan kembali dari transmisi data. *Flow Control* yang biasanya menggunakan :

*None* : Tidak adanya pengontrolan *flow*.

*XON/XOFF* : Karakter *XON/XOFF* akan dikirim oleh penerima ke pengirim. *XON*

dikirim ke pengirim, jika penerima siap menerima data lagi, *XOFF* dikirim ke pengirim untuk berhenti mengirim.

*Hardware* : Pengontrolan diatur oleh *hardware*, menggunakan metode *handshake* antar *hardware* pengirim dan penerima dengan mengirim sinyal.

### 2.3 PC Server

PC Server atau bisa juga disebut *server* adalah sebuah komputer yang menyediakan file, sumber daya, atau layanan tertentu yang dibutuhkan dalam sebuah jaringan. Umumnya komputer yang dijadikan *server* akan memiliki spesifikasi perangkat keras yang lebih tinggi daripada komputer-komputer lain di jaringan tersebut. Sebuah *server* terhubung dengan *client*. Secara mudah, *client* adalah komputer yang bukan *server*. Jika *server* menyediakan file, sumber daya, atau layanan tertentu, maka *client* komputer yang meminta file, sumber daya, atau layanan tersebut [7].

#### 2.3.1 Service-Service Sistem Server

Seperti telah dijelaskan diatas, bahwa *server* menyediakan layanan tertentu (*service-service* sistem) yang dibutuhkan dalam sebuah jaringan. Beberapa *service* itu diantaranya yaitu : *Server Web*, *FTP Server*, *E-mail Server*, *DNS Server*, *Proxy Server*, *Database Server*, *Server File Sharing*, dan sebagainya

#### 2.4 Open Source

*Linux* adalah sistem operasi yang bersifat *multiuser* dan *multitasking*, yang dapat berjalan di *multiplatform*, termasuk pada *platform* prosessor intel 386 maupun yang lebih tinggi. Sistem operasi ini dapat berinteroperasi dengan baik dengan sistem operasi lainnya termasuk *Apple*, *Windows*, *FreeBSD*, dan *Novell*.

*Linux* merupakan pengembangan dari *Minix* (*Mini Unix*), yang dikembangkan oleh *Linux B. Torvalds*. Pada dasarnya, *Linux* adalah sebuah *kernel*, yang merupakan inti dari suatu sistem operasi, yang menangani I/O, pengaturan memori, pengaturan proses dan interaksi antara perangkat keras dan aplikasi. Secara sederhana, untuk sebuah sistem operasi yang lengkap, di atas *kernel* masih ada lapisan *pustaka/system call*, dan lapisan aplikasi.

Oleh sebab itu, *Linux* sering disebut GNU/Linux, mengingat *Linux* tidak dapat berjalan sendiri dan selalu tampil dengan menyertakan program aplikasi tambahan lain untuk masuk di dalamnya. GNU menunjukkan suatu bendera lisensi yang menangani aplikasi-aplikasi yang berjalan di *Linux*.

Pemaketan yang merupakan gabungan antara *kernel* (*Linux*), *pustaka* dan aplikasi yang diracik dan dikompilasi dengan tata cara tertentu itulah yang disebut distribusi *Linux* (*distro*). Beberapa *distro* besar diantaranya *Knoppix*, *Fedora Core*, *Mandriva*, *Debian*, *SuSE*, *Red Hat*, *Slackware*, *Ubuntu* dan sebagainya. *Distro* dapat dikatakan besar apabila memiliki aplikasi-aplikasi yang banyak dan

lengkap. *Distro* tersebut harus memiliki teknologi, ciri khas, basis pengguna serta sangat mungkin memiliki *distro-distro* turunan.

Saat ini *Linux* didistribusikan dengan metode yang bernama *Open Source*, yaitu suatu lisensi perangkat lunak di mana pemilik program tetap memegang hak ciptanya, tetapi orang lain dapat menyebarkan, memodifikasi bahkan menjual kembali program tersebut dengan syarat kode program yang asli harus tetap disertakan.

Richard M Stallman, pendiri *Free Software Foundation* – sebuah organisasi yang mendukung *Open Source*, mengeluarkan lisensi perangkat lunak untuk *Open Source* yang dinamakan *GPL (GNU General Public License)*. Lisensi inilah yang saat ini paling banyak digunakan untuk mendistribusikan *software open source*. Selain *GPL*, masih banyak lisensi *software* lainnya yang dikembangkan oleh komunitas *Open Source* [5].

Dengan metode *open source*, pengguna dapat memperoleh banyak keuntungan seperti, gratis, dapat terlibat dalam pengembangan *software* tersebut sehingga menjadi sesuai yang diinginkan, karena dikembangkan oleh banyak orang, *software* yang dihasilkan akan menjadi benar-benar memiliki kualitas yang baik.

Sedangkan sisi pembuat program pun memiliki keuntungan, karena seluruh komunitas yang menggunakan program tersebut akan dapat membantu untuk membuat perangkat lunak tersebut menjadi lebih baik, tidak ada biaya untuk iklan dan perawatan program karena komunitas itulah yang akan mempromosikan dan mengembangkan program tersebut, metode ini juga dapat digunakan sebagai sarana untuk memperkenalkan konsep baru.

Aspek positif dari *open source* adalah penerimaan yang luas untuk *software* yang memang benar-benar bermutu. *Linux* berkembang sangat pesat dengan menggunakan metode ini.

#### 2.4.1 Knoppix

*Knoppix* merupakan salah satu distribusi besar *Linux* berbasis *Debian*, dengan sebuah terobosan dalam dunia sistem operasi dengan *Linux live CD*-nya. adalah *Knoppix 4.0.2* adalah versi *Knoppix* terbaru yang dipakai dalam proyek akhir ini. *Knoppix 4.0.2* memiliki paketkan aplikasi yang cukup lengkap. Diantaranya adanya aplikasi *tool SMS gateway*, seperti *Kannel*, *Gnokii* dan sebagainya, disertai bahasa pemrograman *Gambas*, *service-service* sistem, seperti *server file sharing (Samba)*, *database server (MySQL)*, *Server web (Apache)*, dan aplikasi lainnya.

#### 2.4.2 Gambas

*Gambas* adalah salah satu bahasa pemrograman yang berorientasi pada grafis atau visual, yang berjalan pada sistem *Linux*. Secara umum, pemrograman ini mirip dengan *Visual Basic* milik Microsoft, namun secara tegas *Benoit Minisini*, pembuat bahasa pemrograman ini, menyatakan *Gambas* tidak sama dengan *Visual Basic* dan tak akan pernah sama [1].

Kemiripan yang terdapat pada *Gambas* dengan *Visual Basic* karena *Gambas* memang dibuat sebagai interpreter bahasa basic, yang juga merupakan cikal bakal *Visual Basic*. Meski demikian, kemiripan yang dimiliki oleh *Gambas* membuat para *programmer* yang biasa bekerja pada *Visual Basic* menjadi lebih mudah untuk migrasi dari *Windows* yang mahal, ke *Linux* yang bersifat *Open Source* dan free ini [1].

Beberapa kelebihan yang dimiliki *Gambas*, dibandingkan dengan *Visual Basic*, diantaranya [1] :

1. *Gambas* bersifat *open source*.
2. Memiliki tampilan *interface GUI* yang *familiar* dan *simple*, sehingga sangat mudah dipelajari.
3. Lingkungan kerja *Gambas* dibuat dengan *Gambas* itu sendiri.
4. *Gambas* dapat dijadikan *debugger* yang baik.
5. Komponen program *Gambas* diletakkan dalam suatu direktori, sehingga bila ingin menghapus *Gambas* dari sistem, bisa dilakukan hanya dengan menghapus direktori tempat *Gambas* yang berada.
6. *Gambas* sangat mudah untuk diterjemahkan ke bahasa apapun.
7. *Gambas* mampu mendukung *database* seperti *PostgreSQL* dan *MySQL*.
8. Saat membuat proyek baru dalam *Gambas*, secara otomatis *Gambas* akan membentuk direktori tersendiri untuk proyek tersebut. Hal ini tidak ditemukan pada *Visual Basic*, kecuali *Visual Basic.Net*.
9. Untuk versi-versi berikutnya, *Gambas* dimungkinkan dapat membaca *source code* dari *Visual Basic*. Indikasi ini dapat dilihat pada pilihan saat akan membuat proyek baru.

#### 2.4.3 Samba

*Samba* merupakan perangkat lunak di *Linux* yang digunakan sebagai *server file sharing*. Biasanya digunakan untuk menghubungkan antara sistem operasi *Linux* dengan sistem operasi *Windows* pada suatu jaringan komputer untuk berbagi file dan printer.

#### 2.4.4 Server web : Apache

*Server web* adalah *software server* yang menjadi tulang belakang dari *world wide web (www)*. *Server web* menunggu permintaan dari client yang menggunakan *browser* seperti *Konqueror*, *Mozilla Firefox*, *Internet Explorer* dan program *browser* lainnya. Jika ada permintaan dari *browser*, maka *server web* akan memproses permintaan itu kemudian memberikan hasil prosesnya berupa data yang diinginkan kembali ke *browser* untuk kemudian di tampilkan sesuai dengan kemampuan *browser* itu dengan format yang standar yaitu *SGML (Standard General Markup Language)*. Pada umumnya, para pengguna internet lebih banyak menggunakan format *HTML (Hypertext Markup Language)* karena penggunaannya lebih sederhana dan mudah di pelajari dibandingkan dengan format lain seperti *SGML*, *PHP*, *CGI*, dan lain-lain [4].

*Server web* untuk berkomunikasi dengan clientnya atau *web browser* mempunyai protokol sendiri yaitu *HTTP (Hypertext Transfer Protocol)*. Dengan protokol ini komunikasi antar *server web* dengan *client*-nya atau *browser* dapat saling di mengerti lebih mudah [4].

Banyak sekali *software server web* yang berada di *internet*. Diantaranya yang paling baik ialah *Apache* karena *server web* ini bersifat *freeware* yang mempunyai dukungan teknis dari pembuatnya dan dengan dokumentasi yang lengkap, mudah mengkonfigurasinya, mempunyai level-level pengamanan dan sebagainya[4].

#### 2.4.5 Command Line Tool

*Command Line Tool* merupakan aplikasi baris perintah di *Linux* yang diketikkan di *prompt* dan diakhiri dengan *enter* untuk mengeksekusi perintah banyak sekali pekerjaan yang bisa dilakukan dengan cepat dan ringkas, seperti perintah *DOS*.

Aplikasi *command line* umumnya [6] :

1. Membutuhkan *resource system* yang relatif rendah, dibandingkan dengan *tool* serupa yang menggunakan *interface* grafis.
2. Relatif tidak membutuhkan sangat banyak pustaka (dibandingkan aplikasi *GUI* sejenis), dan karenanya berukuran cukup kecil sehingga dapat dijalankan dalam kondisi sistem yang ekstrim (misal: dalam *rescue system*).
3. Menyediakan opsi-opsi penggunaan yang menjadikannya sangat fleksibel.
4. Datang dengan tugas-tugas spesifik.
5. Dapat digunakan dalam *shell script*, yang akan sangat membantu untuk tugas-tugas sistem, dan penggunaan yang relatif kompleks.

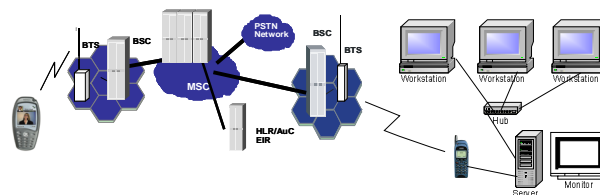
Beberapa contoh *command line tool* adalah sebagai berikut.

- a) *halt* : perintah untuk *off* komputer
- b) *reboot* : perintah untuk *restart* komputer
- c) *su* : perintah untuk masuk sebagai superuser
- d) *man* : perintah untuk melihat penjelasan masing-masing perintah
- e) *ls* : perintah untuk menampilkan file-file
- f) Dan sebagainya

### PERANCANGAN PRORAM SISTEM KENDALI PC SERVER BERBASIS OPEN SOURCE

#### 3.1 Blok Diagram Sistem Kendali PC Server Jarak Jauh

Sistem pengendali *PC Server* jarak jauh dengan menggunakan *SMS gateway* sebagai *interface* dengan *software* untuk menjalankan proses-proses perintah yang dikehendaki oleh *user*. Berikut ini adalah blok diagram sistem kendali *PC server* jarak jauh menggunakan aplikasi *SMS gateway*.



Gambar 3.1 Blok diagram sistem kendali PC Server jarak jauh

Dari blok diagram diatas bisa dilihat, bahwa sistem kendali ini menggunakan jaringan *GSM* untuk bisa bekerja. Oleh sebab itu protokol yang digunakan dalam proyek akhir ini yaitu **Protokol GSM 07.05**. *AT Commands* berdasarkan protokol *GSM 07.05*, digunakan untuk mengoperasikan fungsi *SMS* dari *handphone GSM*. Modul *GSM* dari *handphone GSM* mendukung *SMS mode PDU*, yang digunakan untuk perancangan program dalam proyek akhir ini.

Program yang dirancang ini dapat mengatur *on*, *off* dan *restart service-service* sistem yang ada di *PC Server*. Selain itu program yang dirancang ini juga dapat menjalankan *command line tool*, diantaranya *restart / off PC Server*.

*SMS* yang masuk dari *handphone* pengirim diterima oleh *handphone* terminal. Isi *SMS* tersebut diinisialisasi oleh perintah *AT*,



kemudian mengirimkannya ke *Gambas* untuk dieksekusi sesuai isi perintah di SMS. SMS yang telah tereksekusi kemudian dihapus otomatis oleh perintah AT.

### 3.2 Hardware dan Software Pendukung

#### 3.2.1 Hardware Pendukung

*Hardware* yang digunakan untuk membuat sistem kendali PC Server menggunakan SMS gateway, dibagi menjadi 2 bagian yaitu *hardware* untuk membuat jaringan komputer *client/server* dan *hardware* untuk aplikasi SMS gateway.

##### 3.2.1.1 Hardware Untuk Jaringan Komputer Client/Server

Untuk membuat jaringan komputer *client/server* dibutuhkan *hardware* diantaranya sebagai berikut :

- 2 Minimal 2 buah komputer. Satu sebagai *server* dan lainnya sebagai *workstation*.
- 3 Satu buah *switch*.
- 4 Kabel UTP dan RJ45
- 5 *Crimping tool* dan *cable tester*

##### 3.2.1.2 Hardware Untuk Aplikasi SMS Gateway

Untuk membuat aplikasi SMS gateway dibutuhkan *hardware* diantaranya sebagai berikut :

- HP SIEMENS C55 sebagai perangkat komunikasi terminal yang terhubung ke PC Server.



Gambar 3.2 HP Siemens C55 sebagai terminal (Penerima)

- HP Nokia 6100 sebagai perangkat komunikasi sender.



Gambar 3.3 HP Nokia 6100 sebagai Pengirim

- Kabel data Prolific PL2303 USB sebagai media koneksi ke PC Server.



Gambar 3.4 Kabel Data Prolific PL2303 USB Untuk HP Siemens

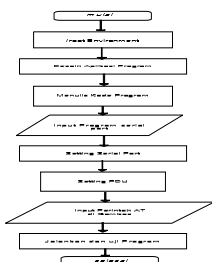
#### 3.2.2 Software Pendukung

Perangkat lunak (*Software*) yang digunakan untuk perancangan sistem kendali PC Server jarak jauh menggunakan SMS gateway berbasis *open source* sebagai berikut:

- ✓ Sistem operasi GNU/LINUX- Knoppix 4.0.2 sebagai PC Server dan Windows 98 sebagai *workstation*.
- ✓ AT Command sebagai tool SMS gateway.
- ✓ Samba sebagai *server file sharing*.
- ✓ Bahasa pemrograman *Gambas* 1.9.20 untuk membuat program sistem kendali PC Server.
- ✓ Apache sebagai *server web*.

### 3.3 Tahap-Tahap Perancangan Program

Tahap-tahap perancangan untuk membuat program sistem kendali PC server jarak jauh menggunakan SMS gateway berbasis *open source* secara umum digambarkan pada diagram alir sebagai berikut :



Gambar 3.5 Diagram alir Tahap-Tahap Perancangan Program

#### 3.4 /root Environment

*/root* merupakan direktori untuk *superuser*, yang berhubungan dengan pengaturan administrasi sistem di *Linux*. Program yang telah dibuat tidak akan berjalan bila tidak berada dalam */root environment*.

Berikut ini adalah urutan proses untuk masuk ke dalam */root environment*.

- 1) Setelah muncul menu *login user* tekan **ctrl+alt+F1** untuk masuk ke dalam *text mode*.
- 2) Login dari user sampai menjadi *superuser*.
- 3) Ketik perintah **ps ax|grep kdm** untuk menampilkan direktori *kdm*
- 4) Matikan direktori */usr/bin/kdm* sesuai kodenya dengan perintah **kill** spasi no.kode.
- 5) Masuk ke dalam */root environment* grafis dengan perintah **startx**

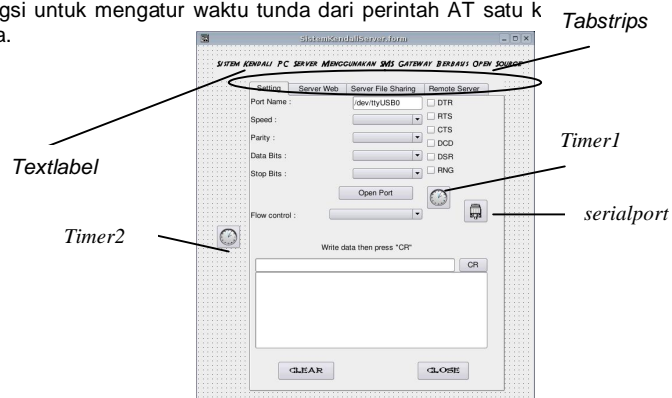
#### 3.5 Perancangan Program Sistem Kendali PC Server Dengan Gambas

##### 3.5.1 Desain aplikasi

Desain aplikasi (program) hanya terdiri dari sebuah *form*. Bagian-bagian dari aplikasi tersebut, dibuat dalam sebuah *tabstrips* yang terpadu.

##### 3.5.1.1 Bagian Form

Bagian *form* merupakan perancangan awal dari desain aplikasi. *Tabstrip* terdiri dari 4 bagian, yaitu bagian *setting*, bagian *server web*, bagian *server file sharing*, dan bagian *remote PC*. Komponen *Serial Port* digunakan untuk menghubungkan perangkat komunikasi (ponsel) dengan komputer melalui *serial port* USB. *Timer* berfungsi untuk mengatur waktu tunda dari perintah AT satu k

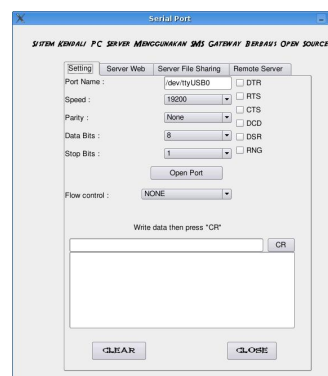


Gambar 3.6 Tampilan Form Menu  
Tabel 3.1 Setting Tiap Komponen Bagian Form

Komponen	Property	Nilai Baru
Form	Name	SistemKendaliServer
Textlabel8	Text	Serial Port
Tabstrip1	Count	4
Timer1	Enabled	FALSE
	Delay	2000
Timer2	Enabled	FALSE
	Delay	2000
SerialPort	Flowcontrol	Hardware
	Parity	None
	Speed	19200
	DataBits	Bits8
	StopBits	Bits1

##### 3.5.1.2 Bagian Setting

Bagian *setting* digunakan untuk men-setting *serial port*. Parameter-parameter *default* yang digunakan untuk komunikasi perangkat komunikasi dengan komputer yaitu *port name* USB, *speed* 19200, *parity* None, *data bits* 8, *stop bits* 1, dan *flow control* None.



Komponen	Property	Nilai Baru
Label	Name	Label2
TextArea	Name	TextArea2
Button	Name	Button8
	Text	Clear
Button	Name	Button12
	Text	Close
Tabstrip	Index	1
	Text	Web Server
Button	Name	Button3
	Text	Apache Start
Button	Name	Button18
	Text	Apache Restart
Button	Name	Button4
	Text	Apache Stop
Frame	Name	Frame1
	Text	APACHE

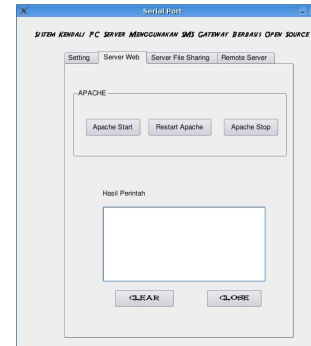
Gambar 3.7 Tampilan Bagian Setting  
Tabel 3.2 Setting Tiap Komponen Bagian Setting

Komponen	Property	Nilai Baru
Textlabel	Text	Port Name :
	Name	TextLabel1
Textlabel	Name	TextLabel2
	Text	Speed :
Textlabel	Name	TextLabel3
	Text	Parity :
Textlabel	Name	TextLabel4
	Text	Data Bits :
Textlabel	Name	TextLabel5
	Text	Stop Bits :
Label	Name	Label1
	Text	Flow control :
Textlabel	Name	TextLabel6
	Text	Write data then press "CR"
TextBox	Name	TxtSend
Button	Name	Button2
	Text	CR
TextArea	Name	TextArea1
Button	Name	Button16
	Text	Clear
Button	Name	Button17
	Text	Close
Tabstrip	Index	0
	Text	Setting
TextBox	Name	TxtPort
	Text	/dev/ttyUSB
CheckBox	Name	ChkDTR
	Text	DTR
CheckBox	Name	ChkRTS
	Text	RTS
CheckBox	Name	ChkCTS
	Text	CTS
CheckBox	Name	ChkDCD
	Text	DCD
CheckBox	Name	ChkDSR
	Text	DSR
CheckBox	Name	ChkRNG
	Text	RNG
Button	Name	Button1
	Text	Open Port
ComboBox	Name	CmbSpeed

ComboBox	Name	CmbParity
ComboBox	Name	CmbData
ComboBox	Name	CmbStop
ComboBox	Name	ComboBox1

### 3.5.1.3 Bagian Server Web

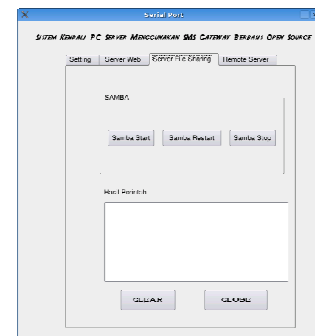
Bagian server web digunakan untuk mengendalikan server web, seperti perintah start, stop dan restart server web. Selain itu bagian ini digunakan untuk mendeteksi jalan tidaknya perintah yang telah dieksekusi dengan melihat di bagian "Hasil Perintah".



Gambar 3.8 Tampilan Bagian Server Web  
Tabel 3.3 Setting Tiap Objek Bagian Server Web

### 3.5.1.4 Bagian Server File Sharing

Bagian server file sharing digunakan untuk mengendalikan server file sharing, seperti perintah start, stop, dan restart server file sharing. Selain itu bagian ini digunakan untuk mendeteksi jalan tidaknya perintah yang telah dieksekusi dengan melihat di bagian "Hasil Perintah".

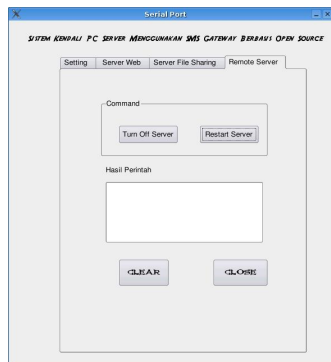


Gambar 3.9 Tampilan Bagian Server File Sharing  
Tabel 3.4 Setting Tiap Objek Server File Sharing

Komponen	Property	Nilai Baru
Label	Name	Label3
TextArea	Name	TextArea3
Button	Name	Button11
	Text	Clear
Button	Name	Button13
	Text	Close
Tabstrip	Index	2
	Text	Server File Sharing
Button	Name	Button9
	Text	Samba Start
Button	Name	Button19
	Text	Samba Restart
Button	Name	Button10
	Text	Samba Stop
Frame	Name	Frame2
	Text	SAMBA

### 3.5.1.5 Bagian Remote PC

Bagian remote PC digunakan untuk mengendalikan PC Server, seperti perintah off dan restart PC Server. Selain itu bagian ini digunakan untuk mendeteksi jalan tidaknya perintah yang telah dieksekusi dengan melihat di bagian "Hasil Perintah".



Gambar 3.10 Tampilan Bagian Remote PC  
Tabel 3.5 Setting Tiap Objek Bagian Remote PC

Komponen	Property	Nilai Baru
TextLabel	Name	TextLabel7
TextArea	Name	TextArea4
Button	Name	Button14
	Text	Clear
Button	Name	Button15
	Text	Close
Tabstrip	Index	3
	Text	Remote PC
Button	Name	Button6
	Text	Turn Off Server
Button	Name	Button5
	Text	Restart Server
Frame	Name	Frame3
	Text	Command

### 3.4.2 Penulisan Kode Program

Prosedur pertama dalam perancangan program sistem kendali PC server yaitu dengan membuat perintah *shell*. Perintah ini digunakan untuk menghidupkan, mematikan atau *re-start service* sistem dan mematikan serta *re-start* PC server.

Tabel 3.6 Persamaan Perintah Shell Dengan Perintah SMS

Perintah SHELL	Perintah SMS	Format PDU
# /etc/init.d/apache start	WEB ON 5153	D7A210F474826AB1DA0C
# /etc/init.d/apache stop	WEB OFF 5153	D7A210F4341A41B5586D06
# /etc/init.d/apache restart	WEB RST 5153	D7A210249D5241B5586D06
# /etc/init.d/samba start	SMB ON 5153	D3A610F474826AB1DA0C
# /etc/init.d/samba stop	SMB OFF 5153	D3A610F4341A41B5586D06
# /etc/init.d/samba restart	SMB RST 5153	D3A610249D5241B5586D06
# reboot	PC RST 5153	D021483AA5826AB1DA0C
# halt	PC OFF 5153	D021E86934826AB1DA0C

Berikut adalah *list* program untuk perintah *shell* diatas.

```

PUBLIC SUB Button3_Click()
    SHELL "/etc/init.d/apache start > sApache" WAIT
    textarea2.Text=textarea2.Text & file.Load("sApache")
END
PUBLIC SUB Button4_Click()
    SHELL "/etc/init.d/apache stop > sApache" WAIT
    textarea2.Text=textarea2.Text & file.Load("sApache")
END
PUBLIC SUB Button9_Click()

```

```

SHELL "/etc/init.d/samba start > sSamba" WAIT
textarea3.Text=textarea3.Text & file.Load("sSamba")
END

PUBLIC SUB Button10_Click()
    SHELL "/etc/init.d/samba stop > sSamba" WAIT
    textarea3.Text=textarea3.Text & file.Load("sSamba")
END

PUBLIC SUB Button5_Click()
    SHELL "reboot > sReboot" WAIT
    textarea1.Text=textarea1.Text & file.Load("sReboot")
END

PUBLIC SUB Button6_Click()
    SHELL "halt > shalt" WAIT
    textarea1.Text=textarea1.Text & file.Load("shalt")
END

PUBLIC SUB Button19_Click()
    SHELL "/etc/init.d/samba restart > sSamba" WAIT
    textarea3.Text=textarea3.Text & file.Load("sSamba")
END

PUBLIC SUB Button18_Click()
    SHELL "/etc/init.d/apache restart > sApache" WAIT
    textarea2.Text=textarea2.Text & file.Load("sApache")
END

```

Prosedur berikutnya memasukkan program *serial port* bersamaan dengan list program perintah *shell* yang telah dibuat.

```

' Gambas class file
'PRIVATE Sport AS SerialPort
'PUBLIC SUB Form_Open()
' Sport=NEW SerialPort AS "Sport"
'END
PUBLIC SUB Form_Close()
    IF Sport.Status=Net.Active THEN CLOSE Sport
END

PUBLIC SUB Check_Status()

    ChkDSR.Value=Sport.DSR
    ChkDTR.Value=Sport.DTR
    ChkCTS.Value=Sport.CTS
    ChkRTS.Value=Sport.RTS
    ChkDCD.Value=Sport.DCD
    ChkRNG.Value=Sport.RNG
END

PUBLIC SUB Button1_Click()
    IF Sport.Status=Net.Active THEN
        CLOSE Sport
        Button1.Text="Open"

    ELSE
        ' Line parameters
        Sport.PortName=TxtPort.Text
        Sport.Speed=CmbSpeed.Text
        Sport.Parity=CmbParity.Index
        Sport.DataBits=CmbData.Text
        Sport.StopBits=CmbStop.Text
        ' keep DTR on
        Sport.FlowControl=ComboBox1.Index
    END IF
END

```



```

Sport.Open()
Check_Status()

TextArea1.Text="Port Opened : " & Sport.PortName & " Settings : " &
Sport.Speed & ", " & Sport.Parity & ", " & Sport.DataBits & ", " &
Sport.StopBits & Chr(13) & Chr(10)

Button1.Text="Close"

```

Prosedur diatas digunakan untuk membuka dan menutup *serial port*, dengan lebih dahulu mengecek status *serial port* pada saat itu aktif atau tidak. Apabila aktif maka proses identifikasi perangkat komunikasi yang terhubung ke komputer melalui *serial port* dilakukan dengan parameter-parameter *default serial port*.

Prosedur berikutnya memasukkan perintah AT untuk menampilkan SMS yang masuk untuk kemudian dibaca dan disimpan ke dalam memori telepon.

```

'Perintah untuk menampilkan SMS yang masuk dan setting SMS yang masuk ke dalam memori
telepon.

PRINT #Sport,"AT+CNMI=1,1,0,0,1" ; Chr$(13); Chr$(10);

WAIT 2
PRINT #Sport,"AT+CPMS=" & Chr(34) & "MT" & Chr(34) & ", " & Chr(34) & "MT" & Chr(34) & ", "&
Chr(34) & "MT" & Chr(34) ; Chr$(13); Chr$(10);

END IF

```

```

ChkRng.Value=iVal
END

```

```

PUBLIC SUB SPort_Read()
DIM s AS String
DIM hasil AS Integer

READ #Sport,s,Lof(Sport)
TextArea1.Text=TextArea1.Text & s

hasil=Instr(textarea1.Text,"+CMTI",1)
IF hasil<>0 THEN
textarea1.Text=""
timer1.Enabled=TRUE
END IF

PUBLIC SUB SPort_DTRChange(iVal AS Boolean)
ChkDTR.Value=iVal
END

PUBLIC SUB SPort_DSRChange(iVal AS Boolean)
ChkDSR.Value=iVal
END

PUBLIC SUB SPort_CTSCChange(iVal AS Boolean)
ChkCTS.Value=iVal
END

PUBLIC SUB SPort_DCDChange(iVal AS Boolean)
ChkDCD.Value=iVal
END

PUBLIC SUB SPort_RTSCChange(iVal AS Boolean)
ChkRTS.Value=iVal
END

```

'Buka Tutup Port Serial USB

```

PUBLIC SUB Button2_Click()

IF Sport.Status = Net.Inactive THEN
Message ("Open port first!")
ELSE
WRITE #Sport,TxtSend.Text & Chr(13) & Chr(10), txtSend.Length + 2
PRINT #Sport,txtSend.Text; Chr$(13); Chr$(10);
END IF

END

PUBLIC SUB ChkDTR_Click()

Sport.DTR=ChkDTR.Value
Check_Status

END

PUBLIC SUB ChkRTS_Click()

Sport.RTS=ChkRTS.Value
Check_Status

END

PUBLIC SUB ComboBox1_Click()

Sport.FlowControl=ComboBox1.Index

END

PUBLIC SUB Form_Open()

cmbSpeed.Index = cmbSpeed.Find("19200")

END

```

'Perintah untuk membaca SMS yang masuk

```

PUBLIC SUB Timer1_Timer()

PRINT #Sport,"AT+CMGR=26" ; Chr$(13); Chr$(10);
timer1.Enabled=FALSE

END

```

Prosedur selanjutnya dilakukan penginputan program di *serial port*.

Masukkan no telepon *default* untuk mengirim perintah SMS. Hanya no telepon ini yang bisa mengeksekusi perintah untuk menjalankan program sistem kendali PC *Server*. No teleponnya yaitu 08176609297.

```

'Setting no.telepon pengirim default untuk mengeksekusi perintah melalui isi SMS
hasil=Instr(textarea1.text,"91261867062979",1)
IF hasil<>0 THEN
WAIT 1

```

SMS yang masuk kemudian diinisialisasi, apakah sesuai antara perintah *shell* dengan perintah SMS. Perintah SMS diubah ke dalam format PDU.

```

'WEB ON
hasil=Instr(textarea1.text,"D7A210F474826AB1DA0C",hasil)
IF hasil<>0 THEN

```

```

Button3_Click
timer2.Enabled=TRUE
textarea1.text=""
END IF

'WEB RST
hasil=Instr(textarea1.text,"D7A210249D5241B5586D06",hasil)
IF hasil<>0 THEN
    Button18_Click
    textarea1.text=""
    timer2.Enabled=TRUE
END IF

'WEB OFF
hasil=Instr(textarea1.text,"D7A210F4341A41B5586D06",hasil)
IF hasil<>0 THEN
    Button4_Click
    textarea1.text=""
    timer2.Enabled=TRUE
END IF

'SMB ON
hasil=Instr(textarea1.text,"D3A610F474826AB1DA0C",hasil)
IF hasil<>0 THEN
    Button9_Click
    textarea1.text=""
    timer2.Enabled=TRUE
END IF

'SMB RST
hasil=Instr(textarea1.text,"D3A610249D5241B5586D06",hasil)
IF hasil<>0 THEN
    Button19_Click
    textarea1.text=""
    timer2.Enabled=TRUE
END IF

'SMB OFF
hasil=Instr(textarea1.text,"D3A610F4341A41B5586D06",hasil)
IF hasil<>0 THEN
    Button10_Click
    textarea1.text=""
    timer2.Enabled=TRUE
END IF

'PC OFF
hasil=Instr(textarea1.text,"D021E86934826AB1DA0C",hasil)
IF hasil<>0 THEN
    Button6_Click
    textarea1.text=""
    timer2.Enabled=TRUE
END IF

'PC RST
hasil=Instr(textarea1.text,"D021483AA5826AB1DA0C",hasil)
IF hasil<>0 THEN
    Button5_Click

```

```

textarea1.text=""
timer2.Enabled=TRUE
END IF

timer2.Enabled=TRUE

END IF
END

```

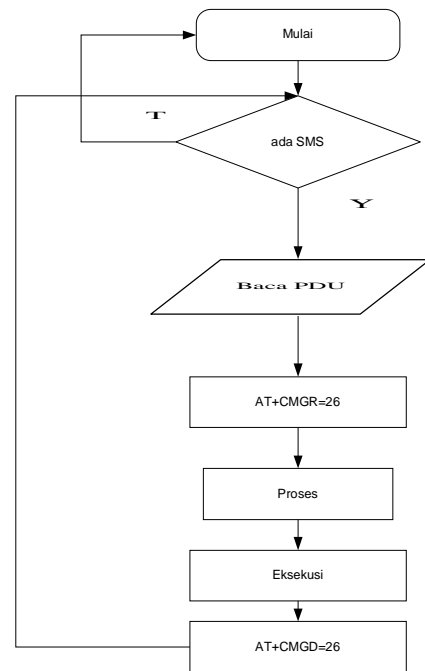
Prosedur terakhir komputer akan otomatis menghapus perintah SMS yang telah tereksekusi di dalam *handphone* dengan perintah AT yaitu AT+CMGD=26. Penghapusan dilakukan karena perintah AT selalu membaca SMS dimulai dari indeks yang pertama.

'Perintah untuk langsung menghapus SMS yang masuk setelah tereksekusi.

```

PUBLIC SUB Timer2_Timer()
    PRINT #Sport,"AT+CMGD=26" ; Chr$(13); Chr$(10);
    timer2.Enabled=FALSE
    textarea1.text=""
END

```



Gambar 3.11 Diagram Alir Program Sistem Kendali PC Server Menggunakan SMS Gateway

## PENGUJIAN DAN ANALISA SISTEM KENDALI PC SERVER BERBASIS OPEN SOURCE

### 4.1 Tujuan Dan Metode Pengujian

Pengujian dilakukan untuk mengetahui apakah program sistem kendali PC server jarak jauh menggunakan SMS gateway dapat bekerja dengan baik. Apabila program tidak dapat bekerja maka dilakukan analisa untuk mengetahui dimana letak kesalahannya untuk kemudian dilakukan perbaikan.

Ada beberapa metode yang digunakan untuk mengetahui sukses tidaknya program sistem kendali PC server jarak jauh menggunakan SMS gateway berbasis open source diantaranya sebagai berikut :

1. Program diuji dengan mengirimkan perintah SMS dari *handphone* ke *terminal*.
2. Hasil dari bekerjanya program dilihat dari pengaruhnya terhadap PC Server dan *client*.
3. Delay (waktu tunda) proses bekerjanya program dihitung menggunakan *stopwatch*.

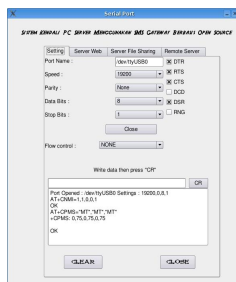
## 4.2 Pengujian Program Sistem Kendali PC Server

### 4.2.1 Setting

Ketika *Button Open Port* ditekan maka komputer membuka *serial port* USB, dan menginisialisasi perangkat komunikasi yang terhubung ke komputer. Akan muncul pemberitahuan di *TextArea* bahwa SMS yang masuk akan dimunculkan, dibaca dan disimpan di memori telepon.

*Serial port* di *setting* berdasarkan parameter-parameter *default* yaitu

*Port name* : `/dev/ttyUSB0`  
*Speed* : `19200`  
*Parity* : `None`  
*Data Bits* : `8`  
*Stop Bits* : `1`  
*Flow Control* : `None`



Gambar 4.1 Tampilan Setting Serial Port Ketika Port Dibuka

### 4.2.2 Server Web

#### 4.2.2.1 Menghidupkan Server Web

Format untuk menghidupkan *server web* yaitu dengan mengetikkan SMS : **WEB(spasi)ON(spasi)5153**.

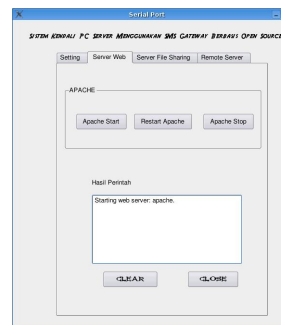
Format penulisan untuk semua perintah SMS pada proyek akhir ini harus dalam bentuk huruf kapital. Karena dalam pembuatan program penulis menggunakan bentuk huruf kapital yang diubah ke dalam format PDU dan kemudian dimasukkan ke dalam penyusunan program. Perubahan format SMS bisa saja dilakukan menjadi huruf kecil atau huruf campuran kapital dan kecil atau berbeda sama sekali dengan format yang dibuat penulis yaitu dengan cara melihat ketentuan perubahan dari format teks biasa menjadi format PDU di Tabel *Ascii*, tanpa mempengaruhi hasil kerja dari sistem kendali PC *Server*.

Program di buat untuk langsung menghapus isi SMS yang tidak sesuai dengan format yang dibuat penulis. Jadi bila format SMS nya tidak sama, maka SMS yang masuk tidak akan dieksekusi dan langsung dihapus oleh program.



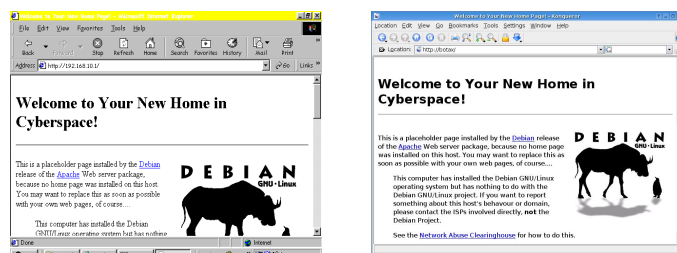
Gambar 4.2 Tampilan SMS Untuk Menghidupkan Server Web Dilaray Handphone

Ketika SMS diterima dan telah diinisialisasi oleh komputer, maka program akan menjalankan *server web* serta melaporkan hasil perintah di *TextArea* bahwa perintah telah berjalan.



Gambar 4.3 Tampilan Bagian Server Web Ketika Apache Dijalankan

Apabila *server web* telah berhasil dijalankan, maka ketika *client* mengakses halaman *web* melalui *web browser* dengan alamat IP *server*, maka akan muncul halaman *web*. Alamat IP *Server* dalam proyek akhir ini yaitu <http://192.168.10.1/>.



Gambar 4.4 Tampilan Web Browser Ketika Apache Telah Dijalankan

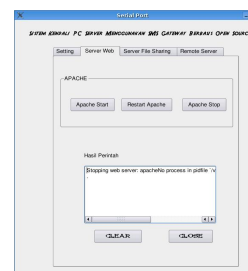
#### 4.2.2.2 Mematikan Server Web

Format untuk mematikan *server web* yaitu dengan mengetikkan SMS : **WEB(spasi)OFF(spasi)5153**.



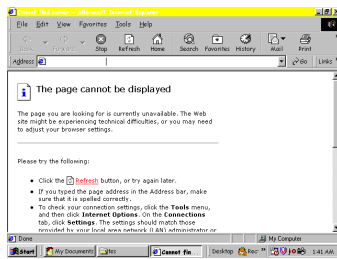
Gambar 4.5 Tampilan SMS Untuk Mematikan Server Web Dilaray Handphone

Ketika SMS diterima dan telah diinisialisasi oleh komputer, maka program akan mematikan *server web* serta melaporkan hasil perintah di *TextArea* bahwa perintah telah berjalan.



Gambar 4.6 Tampilan Bagian Server Web Ketika Apache Dimatikan

Apabila *server web* telah berhasil dimatikan, maka ketika *client* mengakses halaman *web* melalui *web browser* dengan alamat IP *server*, maka tidak akan muncul halaman *web*.



Gambar 4.7 Tampilan Web Browser Ketika Apache Telah Dimatikan

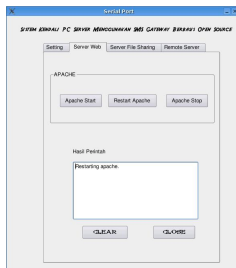
#### 4.2.2.3 Menghidupkan Kembali Server Web

Format untuk me-restart server web yaitu dengan mengetikkan SMS : **WEB(spasi)RST(spasi)5153**.



Gambar 4.8 Tampilan SMS Untuk Merestart Server Web Dilayar Handphone

Ketika SMS diterima dan telah diinisialisasi oleh komputer, maka program akan menghidupkan kembali server web serta melaporkan hasil perintah di *TextArea* bahwa perintah telah berjalan.



Gambar 4.9 Tampilan Bagian Server Web Ketika Apache Dihidupkan kembali

Apabila server web telah berhasil dihidupkan kembali maka ketika *client* mengakses halaman web melalui web browser dengan alamat IP server, maka akan muncul kembali halaman web. Tampilan web browser bisa dilihat pada Gambar 4.4.

#### 4.2.3 Server File Sharing

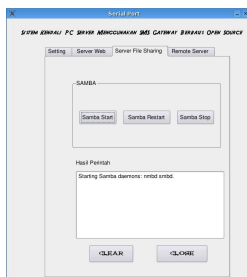
##### 4.2.3.1 Menghidupkan Server File Sharing

Format untuk menghidupkan server file sharing yaitu dengan mengetikkan SMS : **SMB(spasi)ON(spasi)5153**.



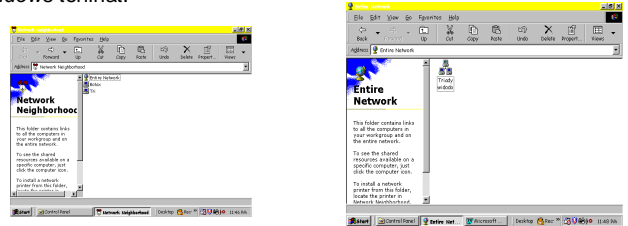
Gambar 4.10 Tampilan SMS Untuk Menghidupkan Server File Sharing Dilayar Handphone

Ketika SMS diterima dan telah diinisialisasi oleh komputer, maka program akan menghidupkan server file sharing serta melaporkan hasil perintah di *TextArea* bahwa perintah telah berjalan.



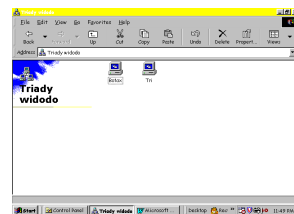
Gambar 4.11 Tampilan Bagian Server File Sharing Ketika Samba Dihidupkan

Apabila server file sharing telah berhasil dihidupkan, maka ketika user mengakses *Network Neighborhood* di *client* (dalam proyek akhir ini *client*nya Windows 98), *workgroup* beserta user Linux dan Windows terlihat.



Gambar 4.12 Jaringan Workgroup Linux-Windows ketika Samba Dijalankan

Pada proyek akhir ini computer name di Linux yaitu **botax**, sedangkan computer name di Windows 98 adalah **Tri**. Nama *workgroup*-nya yaitu **TRIADY WIDODO**.



Gambar 4.13 Computer Name Linux dan Windows 98

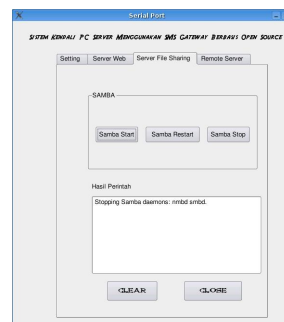
##### 4.2.3.2 Mematikan Server File Sharing

Format untuk mematikan server file sharing yaitu dengan mengetikkan SMS : **SMB(spasi)OFF(spasi)5153**.



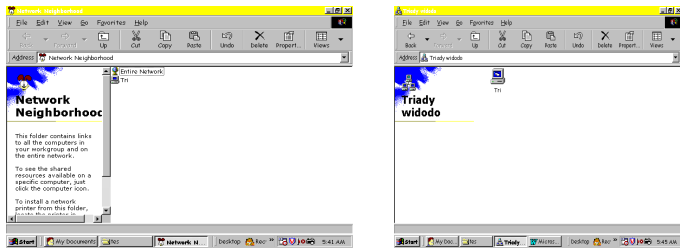
Gambar 4.14 Tampilan SMS Untuk Mematikan Server File Sharing Dilayar Handphone

Ketika SMS diterima dan telah diinisialisasi oleh komputer, maka program akan mematikan server file sharing serta melaporkan hasil perintah di *TextArea* bahwa perintah telah berjalan.



Gambar 4.15 Tampilan Bagian Server File Sharing Ketika Samba Dimatikan

Apabila server file sharing telah berhasil dimatikan, maka ketika user mengakses *Network Neighborhood* di *client* (dalam proyek akhir ini *client*nya Windows 98), user Linux tidak terlihat. Yang terlihat hanya user Windows dan *workgroup*.



Gambar 4.16 Jaringan Workgroup Linux-Windows ketika Samba Dimatikan

#### 4.2.3.3 Menghidupkan Kembali Server File Sharing

Format untuk me-restart server file sharing yaitu dengan mengetikkan SMS : **SMB(spasi)RST(spasi)5153**.



Gambar 4.17 Tampilan SMS Untuk Merestart Server File Sharing Dilayar Handphone

Ketika SMS diterima dan telah diinisialisasi oleh komputer, maka program akan menghidupkan kembali server file sharing serta melaporkan hasil perintah di *TextArea* bahwa perintah telah berjalan.



Gambar 4.18 Tampilan Bagian Server File Sharing Ketika Samba Dihidupkan Kembali

Apabila server web telah berhasil dihidupkan kembali, maka ketika user mengakses *Network Neighborhood* di client (dalam proyek akhir ini clientnya *Windows98*), user *Linux* terlihat kembali. Tampilan *Network Neighborhood* bisa dilihat pada Gambar 4.10 dan Gambar 4.11.

#### 4.2.4 Remote PC

##### 4.2.4.1 Mematikan PC Server

Format untuk mematikan PC Server yaitu dengan mengetikkan SMS : **PC(spasi)OFF(spasi)5153**.



Gambar 4.19 Tampilan SMS Untuk Mematikan PC Server

##### 4.2.4.2 Menghidupkan Kembali PC Server

Format untuk menghidupkan kembali PC server yaitu dengan mengetikkan SMS : **PC(spasi)RST(spasi)5153**.



Gambar 4.20 Tampilan SMS Untuk Merestart PC Server

Ketika SMS diterima dan telah diinisialisasi oleh komputer, maka program akan mematikan atau merestart PC server serta

melaporkan hasil perintah di *TextArea* bahwa perintah telah berjalan. Karena proses yang sangat cepat laporan tidak dapat terlihat dalam waktu yang lama.

#### 4.3 Analisa Program Sistem Kendali PC Server Jarak Jauh

Analisa dilaksanakan dengan memberi perintah SMS dari *handphone* pengirim ke *handphone* penerima (terminal), kemudian dianalisa kecepatan, ketepatan serta kemampuan program untuk menerima instruksi-instruksi dari SMS.

No. Telepon pengirim di set menjadi *default* untuk memberi perintah SMS ke terminal. Hanya no telepon ini yang bisa mengirim perintah melalui SMS ke PC Server. No. telepon pengirim pada proyek akhir ini yaitu **08176609297**. Sedangkan terminal dapat berganti-ganti nomor dengan syarat menggunakan kartu GSM.

Analisa pertama dilakukan untuk menguji kecepatan (dalam detik) tiap-tiap nomor penerima yang berbeda untuk menerima perintah SMS yang dikirimkan *handphone* pengirim.

Tabel 4.1 Hasil Analisa Kecepatan Menerima Perintah Pada Tiap-Tiap Kartu GSM

Kartu GSM Penerima	Delay Operator ( s )	Server Web ( s )			Server File Sharing ( s )			Remote PC ( s )	
		Start	Stop	Restart	Start	Stop	Restart	Turn Off	Restart
XL	7	6	6	7	6	5	6	5	6
IM3	8	7	6	7	6	7	6	6	7

Analisa :

Dapat dilihat dari tabel 4.1 bahwa kartu GSM mempengaruhi kecepatan dari instruksi yang dijalankan. Waktu *delay* terlama ada dalam proses pengiriman SMS dari *handphone* pengirim ke *handphone* penerima. Proses *delay* operator penulis tidak membahasnya dalam proyek akhir ini. Dalam pengujian diatas ternyata waktu *delay* tercepat yaitu kartu GSM Simpati dengan waktu delay 5 detik dan delay terlama yaitu kartu GSM 3 dengan waktu delay 10 detik. Ketika SMS telah diterima *handphone* terminal, maka proses komputer akan lebih cepat dalam mengeksekusi perintah SMS. Rata-rata waktu delay yang terjadi dalam proses eksekusi yaitu 6 detik. Terjadi perbedaan waktu *delay* antara perintah *start*, *stop* atau *restart*, dikarenakan adanya proses inisialisasi yang berbeda terhadap isi SMS yang dibaca dalam format PDU sebelum dieksekusi.

Analisa selanjutnya adalah menguji ketepatan dan kemampuan perintah SMS yang masuk ke *handphone* terminal.



Program akan mengeksekusi perintah SMS yang benar seperti tabel di bawah ini. Apabila tidak sesuai maka isi SMS tersebut langsung dihapus oleh program.

Tabel 4.2 Tabel Perintah SMS Yang Benar

Perintah SMS	Perintah SHELL yang Dijalankan
WEB ON 5153	# /etc/init.d/apache start
WEB OFF 5153	# /etc/init.d/apache stop
WEB RST 5153	# /etc/init.d/apache restart
SMB ON 5153	# /etc/init.d/samba start
SMB OFF 5153	# /etc/init.d/samba stop
SMB RST 5153	# /etc/init.d/samba restart
PC RST 5153	# reboot
PC OFF 5153	# halt

Analisa :

Dapat dilihat pada tabel 4.2, terlihat bahwa SMS yang masuk harus sesuai dengan format di atas untuk menjalankan instruksi-instruksi dalam program sistem kendali PC server.

Format isi SMS dalam proyek akhir ini tidak *case sensitive* yaitu harus dalam bentuk kapital. Contoh apabila kita ingin menghidupkan *server web* dengan perintah **Web on 5153**, maka perintah tersebut tidak akan dijalankan melainkan langsung dihapus oleh program dengan perintah AT+CMGD=26.

## KESIMPULAN DAN SARAN

### 5.1 Kesimpulan

1. SMS Gateway dapat digunakan untuk mengkomunikasikan antara sistem operasi komputer, dengan perangkat komunikasi yang terpasang untuk menjalankan aplikasi *remote control system*.
2. Open Source sebagai suatu lisensi perangkat lunak *free* terbukti handal untuk menjalankan aplikasi sistem kendali PC server jarak jauh menggunakan SMS gateway.
3. Gambas ialah perangkat lunak yang digunakan untuk membuat program aplikasi komputer dengan tampilan grafis (GUI = *Graphical User Interface*) di Linux dapat digunakan untuk mengakses *port serial* USB, yang digunakan untuk membuat aplikasi SMS gateway.
4. AT Command merupakan tool SMS gateway yang dapat digunakan untuk komunikasi dengan *Serial Port* komputer dengan mengirimkan perintah-perintah AT.
5. Protocol Data Unit merupakan salah satu format pengiriman dan penerimaan SMS dalam bentuk oktet heksadesimal dan oktet semidesimal dengan panjang pesan mencapai 160 karakter (7 bit) atau 140 karakter (8 bit).

### 5.2 Saran

1. Aplikasi SMS gateway *remote control system* dapat digabungkan dengan sistem *autoresponder*, untuk melaporkan hasil dari perintah yang telah dikerjakan.
2. Pemakaian tool SMS gateway seperti *Gnokii*, *Kannel* dan sebagainya dapat dijadikan alternatif untuk kemudahan membuat aplikasi SMS gateway lainnya.
3. Sistem Kendali PC Server dapat dihubungkan dengan database server MySQL atau PostgreSQL untuk kemudahan manajemen database.
4. Sistem monitoring jaringan merupakan pengembangan dari sistem kendali PC server untuk kemudahan pengendalian dan troubleshooting jaringan komputer.

## DAFTAR PUSTAKA

- [1] Yudianto. *Membangun Aplikasi SMS Gateway Di Linux*. Dian Rakyat, Jakarta. 2006.
- [2] Ridho, Mahdi. *Pemrograman Gambas, Pemrograman ala Visual Basic di Linux*. Penerbit ANDI, Yogyakarta. 2006.
- [3] Linux Tanpa Instalasi dengan Knoppix. Penerbit ANDI, Yogyakarta. 2005.
- [4] Purbo, Onno W., Dodi Maryanto., Widjil Widodo., dan Syahril Hubbany. *Membangun Server Internet dengan FreeBSD*. Elex Media Komputindo, Jakarta. 2000.
- [5] Prakoso, Samuel. *Jaringan Komputer Linux Konsep Dasar, Instalasi, Aplikasi, Keamanan, dan Penerapan*. Penerbit ANDI, Yogyakarta. 2005.
- [6] Noprianto. *Panduan Praktis Debian GNU/Linux 3.1*. Dian Rakyat, Jakarta. 2006.
- [7] Kurniawan, Yahya. *Kiat Jitu Membangun Jaringan Linux Dengan Windows*. Elex Media Komputindo, Jakarta. 2005.