TUGAS AKHIR – TE 141599

TESTBED KOMUNIKASI TERINTEGRASI UNTUK ANGKUTAN MASSAL CEPAT SURABAYA

TIFFANY MALIATI KHUMAIROH AFANDI
NRP 2212 100 182

Dosen Pembimbing
Dr. Ir. Achmad Affandi, DEA.
Ir. Djoko Suprajitno Rahardjo, MT.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT – TE 141599

Integrated Communications Testbed for Surabaya Mass Rapid Transport

TIFFANY MALIATI KHUMAIROH AFANDI
NRP 2212 100 182

Lecture Advisor
Dr. Ir. Achmad Affandi, DEA.
Ir. Djoko Suprajitno Rahardjo, MT.

ELECTRICAL ENGINEERING MAJOR
Industrial Technology Faculty
Sepuluh Nopember Institute of Technology

Surabaya 2016

***ESTBED* KOMUNIKASI TERINTEGRASI UNTUK ANGKUTAN MASSAL CEPAT SURABAYA**

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Telekomunikasi Multimedia
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui,

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Ir. Achmad Affandi, DEA	**Ir. Djoko Supraitno R, MT.**
NIP. 196510141990021001	NIP. 195506221987011001

SURABAYA,
MEI, 2016

TESTBED KOMUNIKASI TERINTEGRASI UNTUK ANGKUTAN MASSAL CEPAT SURABAYA

Nama : **Tiffany Maliati Khumairoh Afandi**
Pembimbing : **Dr. Ir. Achmad Affandi, DEA**
Ir. Djoko Suprajitno R, MT.

ABSTRAK

Pemanfaatan teknologi informasi sebagai sarana pengelolaan Kota Surabaya dapat meningkatkan keefektifan pelayanan publik. Salah satu caranya adalah dengan penggunaan *On-Board Unit* (OBU) pada Angkutan Massal Cepat yang akan diterapkan di Kota Surabaya. OBU adalah terminal elektronik yang dapat memberikan informasi mengenai identitas dan posisi AMC, dan menghubungkan komunikasi antara CC-ROOM dengan pengemudi. Tugas akhir ini bertujuan untuk merancang sebuah sistem yang digunakan pada *testbed* komunikasi yang terintegrasi antara OBU (*client)* dengan CC-ROOM (*server*).

Perancangan dan implementasi dilakukan dalam 2 tahap, yaitu membuat arsitektur jaringan dan aplikasi sistem *client-server* yang digunakan untuk sistem komunikasi terintegrasi AMC. Aplikasi pengiriman data dari *client* ke *server* dibuat dengan menggunakan Borland Delphi 7. Jaringan yang digunakan yaitu jaringan seluler dan jaringan *backbone.*

Pengujian sistem komunikasi terintegrasi AMC dilakukan dengan 2 macam pengujian, yaitu pengujian sistem *client-server* dan pengujian jaringan sistem yang terdiri dari *packet loss* dan *throughput*. Dari hasil pengujian *packet loss* dan *throughput* didapatkan rekomendasi ukuran data yang dapat diterapkan pada sistem komunikasi terintegrasi angkutan massal cepat Surabaya, yaitu sebesar 450 *byte.*

Kata kunci : *Intelligent Transportation System* , *Packet Loss, Testbed, Throughput*

INTEGRATED COMMUNICATIONS TESTBED FOR SURABAYA MASS RAPID TRANSPORT

Name : Tiffany Maliati Khumairoh Afandi
Supervisors : Dr. Ir. Achmad Affandi, DEA
Ir. Djoko Suprajitno R, MT.

ABSTRACT

The utilization of information technology as a means of managing the city of Surabaya can improve the effectiveness of public services. One of the ways is with the use of on-board units (OBU) at Rapid mass transit that will be applied in the city of Surabaya. OBU is an electronic terminal that can provide the information of the identity and the position of the transportation, and the communication between CC-ROOM and the driver. The purpose of this final project is to design a system that is used for testbed of integrated communication between OBU (client) with CC-ROOM (server).

The design and the implementation had been done in two stages, which to create a network architecture and an application of client-server system that is used for communication system integrated AMC. The application of data transmission from the client to the server was created by using Borland Delphi 7. The network that were used are cellular network and backbone network.

Testing communications systems integrated AMC was done with 2 kinds of testing, which are testing system for client-server and testing system networks. Testing system network consists of packet loss and throughput. From the results of testing packet loss and throughput obtained recommendations datasize that can be applied to an integrated communication system of Surabaya mass rapid transport, that amounted to 450 bytes.

Keywords : *Intelligent Transportation System , Packet Loss, Testbed, Throughput*

KATA PENGANTAR

Segala Puji serta syukur Kepada Allah SWT, karena atas limpahan kasih sayang dan rahmat-Nya saya dapat menjalani dan menyelesaikan tugas akhir yang berjudul :

***Testbed* Komunikasi Terintegrasi untuk Angkutan Massal Cepat Surabaya**

Tugas akhir ini merupakan salah satu syarat yang harus dipenuhi dalam menyelesaikan program studi strata 1 pada Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya.

Tentunya banyak orang yang berjasa dalam proses pengerjaan tugas akhir ini. Terima kasih pertama saya ucapkan kepada orang tua saya yang selalu mendoakan dalam setiap langkah dan selalu memberi semangat. Kedua, kepada Bapak Dr. Ir. Achmad Affandi, DEA, Bapak Ir. Djoko Suprajitno R, MT, dan Bapak Didit selaku dosen pembimbing yang selalu memberi bantuan dan masukan demi terselesaikannya tugas akhir ini. Ketiga, kepada dosen-dosen ITS yang telah mengamalkan ilmunya dalam proses belajar mengajar, semoga ilmu yang ditularkan akan senantiasa bermanfaat bagi penulis dan lingkungan sekitar penulis. *Last but not least*, terima kasih kepada teman-teman seperjuangan selama kuliah di Teknik Elektro ITS yang senantiasa memberikan bantuan baik melalui diskusi maupun semangat serta canda tawa selama proses pengerjaan tugas akhir ini.

Hasil dan manfaat dari penelitian ini diharapkan dapat digunakan untuk pengembangan metode atau teknik lainnya yang bermanfaat untuk kemajuan umat manusia. Dalam proses perbaikan dan pengembangan, kritik dan saran terhadap penelitian ini sangat dibutuhkan untuk mengetahui kekurangan dan kebutuhannya. Terima kasih.

Surabaya, Juli 2016

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kota Surabaya mengalami peningkatan jumlah kendaraan pribadi yang besar dari tahun ke tahun sehingga dapat menyebabkan kepadatan lalu lintas di ruas jalan raya Surabaya. Salah satu penyebabnya adalah kurangnya moda transportasi publik yang memadai. Badan Perencanaan Pembangunan Nasional (Bappenas) telah memperkirakan Angkutan Massal Cepat (AMC) Kota Surabaya beroperasi pada tahun 2017. Proyek AMC Kota Surabaya telah masuk dalam Rancangan Program Jangka Menengah Nasional (RPJMN) Bappenas tahun 2015-2019. Hal ini menunjukkan bahwa pemerintah Kota Surabaya telah berkomitmen untuk membangun sebuah Angkutan Massal Cepat (AMC) untuk mengurangi kemacetan lalu lintas di Kota Surabaya.

Angkutan Massal Cepat (AMC) memiliki sistem komunikasi terintegrasi dengan wujud pemasangan *On-Board Unit* (OBU) pada setiap armada dan sistem *Passenger Information Board* (PIB) pada halte. Data dari OBU akan melewati jaringan seluler sedangkan data dari PIB akan melewati jaringan *backbone*. Kemudian data tersebut akan diterima oleh *Control Center Room* (CC-ROOM). Untuk menguji pengiriman data di perlukan suatu *testbed* terhadap *prototype* dari sistem tersebut. Maka dibuatlah sebuah aplikasi yang mampu mesimulasikan *testbed* pada sejumlah titik OBU dan PIB.

1.2 Perumusan Masalah

Permasalahan yang dibahas dalam tugas akhir ini adalah :

1. Bagaimana rancang bangun *testbed* untuk sejumlah titik *On-Board Unit* (OBU) dan sistem *Passenger Information Board* (PIB) angkutan massal cepat Surabaya?
2. Bagaimana mendapatkan rekomendasi ukuran data terbaik untuk sistem komunikasi terintegrasi angkutan massal cepat Surabaya?

1.3 Batasan Masalah

Hal-hal yang akan dilakukan dalam penelitian ini adalah sebagai berikut:

1. Aplikasi yang dibuat berfungsi sebagai *client*

2. Perangkat lunak yang akan digunakan untuk membagun aplikasi ini adalah Borland Delphi 7
3. Pengujian pengiriman paket data akan dilakukan dengan menggunakan jaringan wifi dan jaringan seluler.
4. Pengujian ini dilakukan menggunakan *server* yang terdapat pada its2.lawanghosting.pw yang memiliki IP *Public*
5. *Packet loss* terkecil
6. *Throughput* terbaik

1.4 Tujuan

Tujuan utama dari penelitian ini adalah membangun sebuah rancangan yang mampu melakukan *testbed* terhadap sistem *prototype* Angkutan Massal Cepat (AMC) dengan menggunakan *server* its2.lawanghosting.pw. Sehingga dapat mengetahui bahwa data yang dikirim dari OBU dan PIB ke CC-ROOM berhasil.

1.5 Metodologi

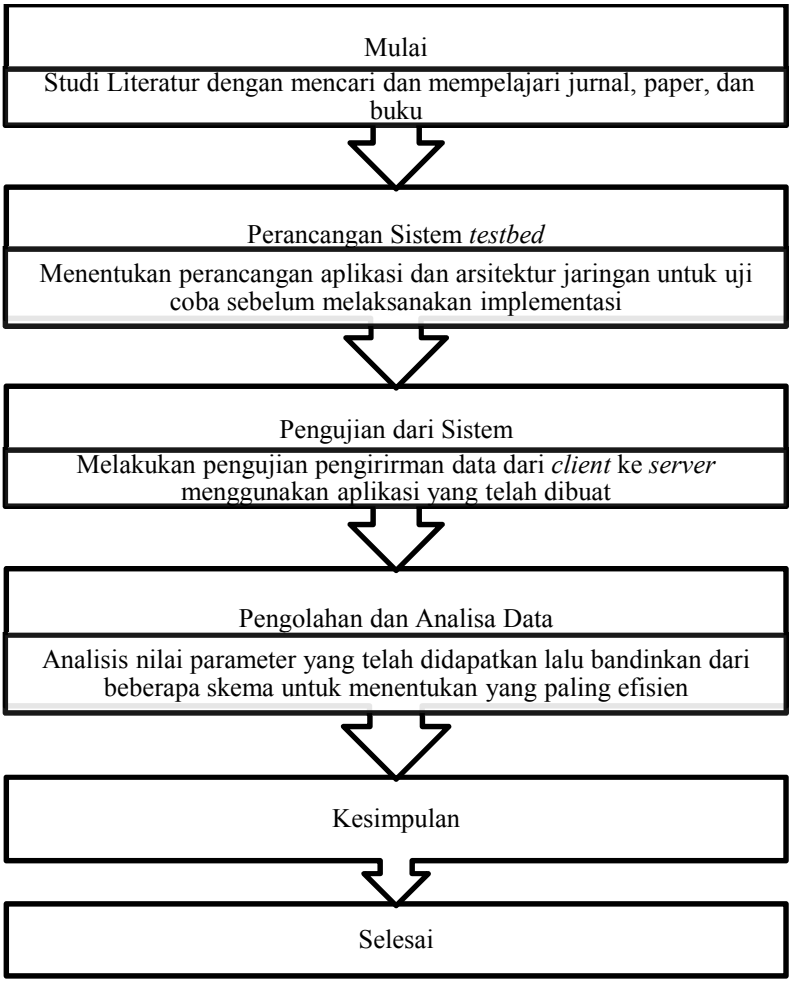
Secara sederhana, dari uraian diatas maka metodologi yang digunakan adalah studi literatur, perancangan simulasi, simulasi, analisis, dan kesimpulan.

Studi literatur dilakukan dengan mencari dan mempelajari beberapa buku, paper, dan jurnal baik skala nasional maupun internasional yang sekiranya dapat menunjang tugas akhir ini. Pada tahap ini akan dipelajari cara kerja dari sistem dan arsitektur jaringan dari sistem komunikasi terintegrasi.

Tahap selanjutnya melakukan perancangan simulasi *testbed* menggunakan perangkat lunak Borland Delphi 7. Pada tahap ini akan diterapkan simulasi berupa uji coba skema-skema yang akan digunakan dengan perangkat lunak.

Setelah melakukan simulasi sistem pada perangkat lunak yaitu akan uji coba sistem tersebut dan mengumpulkan data-data yang dibutuhkan. Skema-skema yang telah dirancang akan diuji coba dan dibandingkan hasilnya.

Setelah mendapatkan hasil data dari 3 skema pengujian yang berbeda, maka akan dilakukan pembandingan dan pengolahan data dari kedua hasil data terebut. Sehingga nantinya data tersebut bisa dianalisa dan akan dihasilkan kesimpulan dari penelitian tugas akhir ini.


Mulai
Studi Literatur dengan mencari dan mempelajari jurnal, paper, dan buku
Perancangan Sistem *testbed*
Menentukan perancangan aplikasi dan arsitektur jaringan untuk uji coba sebelum melaksanakan implementasi
Pengujian dari Sistem
Melakukan pengujian pengirirman data dari *client* ke *server* menggunakan aplikasi yang telah dibuat
Pengolahan dan Analisa Data
Analisis nilai parameter yang telah didapatkan lalu bandinkan dari beberapa skema untuk menentukan yang paling efisien
Kesimpulan
Selesai

1.6 Sistematika Penulisan

Sistematika penulisan pada Tugas Akhir ini dibagi menjadi beberapa bab dengan rincian :

BAB I PENDAHULUAN

Menguraikan latar belakang, perumusan masalah, batasan masalah, tujuan dan manfaat yang berkaitan dengan pengerjaan dan penyusunan Tugas Akhir ini.

BAB II TINJAUAN PUSTAKA

Pada bab ini akan dibahas landasan teori mengenai konsep *Intelligent Transportation System,* angkutan masal cepat, konsep *testbed* sistem komunikasi *client-server*, protokol komunikasi untuk pengiriman paket data dari *client-server*, dan QoS (*Quality of Service*) jaringan.

BAB III PERANCANGAN DAN IMPLEMENTASI SISTEM

Pada bab ini akan dijelaskan tentang perancangan dan implementasi sistem *client-server* yang digunakan untuk melakukan *testbed* komunikasi terintegrasi AMC Surabaya. Sistem *client-server* di rancang dan di implementasikan menggunakan *software* Borland Delphi 7. Jaringan komunikasi *client-server* dirancang dan di implementasikan menggunakan jaringan wifi dan jaringan seluler.

BAB IV PENGUJIAN DAN ANALISIS

Pada bab ini berisi hasil dari rancangan yang sudah di buat pada bab III. Dari pengujian ini kemudian dianalisis dan ditarik kesimpulan sementara mengenai parameter-parameter yang telah diuji

BAB V PENUTUP

Pada bab ini berisi kesimpulan dari keseluruhan materi dan dari hasil analisis data pada bab IV. Selain itu pada bab ini dibahas mengenai saran yang bisa dilakukan untuk pengembangan penelitian selanjutnya

1.7 Manfaat

Manfaat dari tugas akhir ini adalah :

1. Dapat dijadikan sebagai acuan *testbed* pada sistem transportasi cerdas khususnya pada aplikasi *tracking* dan *monitoring*.
2. Memberi konstribusi pada pengembangan teknologi transportasi cerdas.

3. Meningkatkan kinerja operasional , layanan, dan persepsi terhadap angkutan umum kepada konsumen.

BAB II
TINJAUAN PUSTAKA

2.1 *Intelligent Transportation System*

Intelligent Transportation System merupakan sistem tranportasi yang mengombinasikan antara komunikasi, komputer dan teknologi pengawasan yang telah dikembangkan dan diterapkan pada daerah transportasi untuk meningkatkan sistem performansi, keamanan, efisiensi, produktivitas, tingkat layanan, dampak lingkungan, konsumsi energi, dan mobilitas.

Sistem transportasi cerdas dibagi menjadi tiga bidang :

1. Infrastruktur cerdas (*intelligent infrastructure*), contohnya sinyal lalu lintas di jalan raya, tanda pesan variable untuk mengingatkan pengguna jalan dari bahaya yang terjadi di depan, dan sinyal *ramp* jalan bebas hambatan yang bekerja untuk menjaga kelancaran jalan raya
2. Kendaraan cerdas (*smart vehicle*), seperti notifikasi tabrakan otomatis, pemberitahuan batas kecepatan, peringatan tabrakan mudur dan depan, sistem navigasi GPS, dan *interlock* pengapian alkohol
3. Layanan informasi (*information services*), contohnya informasi kedatangan bus pada ponsel, sistem navigasi dalam mobil yang menerima kondisi lalu lintas terkini untuk panduan kemacetan, dan program akses nasional untuk truk.[3]

2.1.1 *Advanced Public Transportation System (APTS)*

Sistem transportasi publik canggih, memungkinkan moda transpor publik seperti kereta, trem dan bus dapat melaporkan posisi mereka (*automatic* vehicle location, AVL), sehingga penumpang mendapat informasi status *real-time* armada dan bagi pengelola dapat memonitor keberadan aset mereka. APTS dapat meningkatkan keamanan, keandalan, efisiensi kepadatan, dan juga dapat mengurangi biaya.[3]

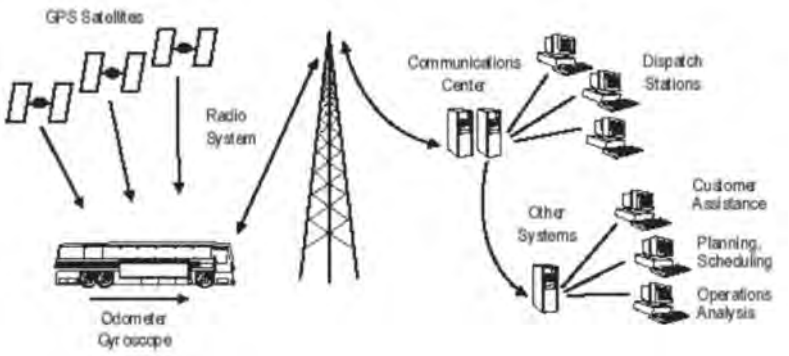
APTS memiliki beberapa jenis, yaitu :

1. Sistem prioritas sinyal
2. Informasi *real-time* untuk penumpang
3. Sistem *automatic vehicle location system*
4. *Computer assisted dispatching systems*

5. Komunikasi radio suara dan data
6. *Scheduling and run cutting systems*
7. *Fare collection systems*
8. *Rapid bus transit systems*
9. Sistem keamanan dan perlindungan
10. *Light rail transit grade crossing warning systems*

2.1.2 *Automatic Vehicle Location (AVL)*

Automatic vehicle location (AVL) adalah sistem pelacak (*tracking*) kendaraan berbasis komputer. Posisi terkini setiap kendaraan akan termonitor dan akan diteruskan menuju pusat kendali (*control center*). Umumnya, informasi posisi kendaraan akan disimpan pada kendaraan yang berlangsung selama beberapa detik. Informasi posisi dapat diteruskan menuju pusat kendali dalam bentuk data mentah atau setelah diproses oleh perangkat yang ada pada kendaraan sebelum ditransmisikan.



Gambar 2.1 Prinsip Kerja AVL

Teknologi yang telah banyak digunakan saat ini untuk melacak kendaraan pada sistem AVL adalah GPS (*Global Position System*). Teknologi GPS menggunakan sinyal yang ditransmisikan dari jaringan 24 satelit yang mengorbit mengelilingi bumi.

Berdasar dari kondisi sarana dan prasarana serta infrastruktur yang sudah ada saat ini, maka dibutuhkan sebuah desain sistem manajemen armada yang berbasis TIK untuk Kota Surabaya.[5]

2.2 Angkutan Massal Cepat

Angkutan Massal Cepat adalah sebuah sistem transportasi umum transit cepat yang sedang di bangun di Surabaya. Dalam upaya mengatasi permasalahan transportasi di Surabaya diantaranya yaitu kemacetan, keterbatasan lahan untuk pembangunan jalan, perjalanan penumpang yang tidak efisien, dan meningkatnya polusi udara atas penggunaan kendaraan pribadi maka dibutuhkan upaya penyediaan angkutan umum massal yang merupakan sarana transportasi berkelanjutan. Di sisi lain, ketersediaan Angkutan Massal Cepat menghubungkan prasarana transportasi utama di Kota Surabaya diharapkan mampu meningkatkan mobilitas orang secara efisien terhadap wilayah Surabaya dan sekitarnya.

Tujuan dikembangkannya Angkutan Massal Cepat adalah :

- Meningkatkan kualitas layanan dan keselamatan lalu-lintas
- Menurunkan tingkat kemacetan
- Meningkatkan efisiensi transportasi
- Mengurangi polusi udara
- Meningkatkan efisiensi pemanfaatan energi

Sistem angkutan massal yang umum dipergunakan di perkotaan adalah sebagai berikut :

1. *Bus Rapid Transit* (BRT) - Teknologi berbasis Bis, pada umumnya beroperasi pada jalur khusus yang sebidang dengan permukaan jalan yang ada, pada kondisi tertentu (misalnya, persimpangan atau pusat kota) yang diperlukan pemisahan elevasi, BRT dilewatkan terowongan atau jembatan khusus.
2. *Light Rail Transit* (LRT) - Teknologi berbasis Rel-Listrik, pada umumnya beroperasi menggunakan kendaraan rel tunggal atau kereta listrik pendek di jalur rel khusus sebidang dengan permukaan tanah dengan konektor listrik di atas kendaraan. Jenis lain dari LRT adalah Tram System, pada umumnya dengan ukuran kendaraan yang lebih kecil dan beroperasi di jalur jalan raya tanpa pemisahann dengan lalu-lintas lainnya.
3. *Underground Metro* - Teknologi berbasis kereta api (*heavy rail*) beroperasi pada jalur di bawah permukaan tanah atau terowongan.
4. *Elevated Rail Transit* - Teknologi berbasis kereta api (heavy *rail*) beroperasi pada jalur di atas permukaan tanah atau jalan layang.
5. *Suburban Rail* - Teknologi berbasis kereta api yang beroperasi pada jalur khusus di permukaan tanah atau di atas permukaan

tanah, pada umumnya melayani pernumpang dari pinggiran kota ke kota.

6. *Personal Rapid Transit* (PRT) - Teknologi berbasis rel atau roda, mengangkut penumpang dengan kendaraan berfasilitas AVG (*automatic guided vehicles*) yang beroperasi pada jalur khusus.[2]

2.3 *Client-Server*

Client-server merupakan teknologi pendistribusian kerja aplikasi antara dua komputer atau lebih yang dihubungkan oleh jaringan komunikasi, dimana yang satu akan bertindak sebagai *server* atau pemberi layanan dan yang lainnya sebagai *client* atau peminta layanan. Baik *client* maupun *server* memiliki pemroses atau CPU sendiri sedangkan jaringan yang digunakan bisa berupa jaringan lokal (LAN) ataupun jaringan yang lebih luas (WAN).[1]

Karakteristik *server* :

1. Pasif
2. Menunggu *request*
3. Menerima *request*, memproses dan mengirimkan balasan berupa *service*

Karakteristik *client* :

1. Aktif
2. Mengirim *request*
3. Menunggu dan menerima balasan dari *server*

Server adalah komputer yang dapat memberikan *service* ke *client*, sedangkan *client* adalah komputer yang mengakses beberapa *service* yang ada di *server*. Ketika *client* membutuhkan suatu *service* yang ada di *server*, *client* akan mengirim *request* kepada *server* lewat jaringan. Jika *request* tersebut dapat dilaksanakan, maka *server* akan mengirim balasan berupa *service* yang dibutuhkan untuk saling berhubungan dengan menggunakan *socket*.

Gambar 2.2 Arsitektur *Client-Server*

2.3.1 Socket

Struktur *Client-server* memungkinkan untuk berbagi informasi dan sumber daya antar sistem, misalnya file, *disk space*, prosesor, dan *peripheral* dapat berkolaborasi dan menyampaikan pesan pasti antara prosesor. Beberapa prosesor dijalankan sebagai *client* dan ada yang sebagai *server*. Harus ada semacam mekanisme yang membuat setiap prosesor dapat mengetahui proses berjalannya jaringan mesin lain ketika dua prosessor sedang berkomunikasi melalui jaringan.

Socket menyediakan *interface* pemrograman sederhana yang memiliki hubungan erat dengan proses komunikasi single dan multi mesin. Tujuan dari *socket* adalah untuk menyediakan sebuah metode umum IPC pada *transport layer*, apakah prosesornya berjalan pada mesin yang sama.

Aplikasi *Socket* terdiri dari dua kategori berdasarkan pengiriman datanya, yaitu :

1. Datagram *Socket* (menggunakan UDP)
2. *Stream Socket* (menggunakan TCP).[6]

2.3.1.1 Socket UDP

Tugas akhir ini menggunakan *socket* UDP sebagai mekanisme komunikasi pertukaran datanya. UDP tidak memerlukan proses koneksi terlebih dahulu untuk mengirimkan data, paket-paket data yang dikirimkan UDP bisa jadi melalui rute yang berbeda-beda, sehingga hasil yang diterima bisa jadi tidak berurutan.

Aplikasi *socket* yang menggunakan UDP lebih memprioritaskan pada pengumpulan data, tidak menggunakan *field sequence* dan ACK, *byte* tambahan yang lebih sedikit (ukuran data lebih kecil), tidak diperlambat oleh proses penerimaan, dan memori dapat dibebaskan lebih cepat sehingga *server* tidak terbebani. Oleh karena itu, aplikasi *socket* dengan UDP cocok diterapkan untuk aplikasi monitoring jaringan, *game online,* dan aplikasi-aplikasi *broadcast*.

2.4 Jaringan Seluler

Jaringan seluler menjangkau area yang jauh lebih luas. Jangkauan umumnya mencangkup nasional dengan infrastruktur jaringan *wireless* yang disediakan oleh *wireless service carrier* untuk biaya pemakaian bulanan, mirip dengan langganan ponsel. Untuk transmisi data menggunakan sistem CDMA, GSM, EDGE, 3G, dan HSPDA.

Jaringan *wireless* beroperasi dalam sebuah jaringan yang membagi kota atau wilayah kedalam sel-sel yang lebih kecil. Satu sel mencakup beberapa blok kota atau sampai 250 mil persegi. Setiap sel menggunakan sekumpulan frekuensi radio atau saluran-saluran untuk memberikan layanan di area spesifik. Kekuatan radio ini harus dikontrol untuk membatasi jangkauan sinyal geografis. Oleh Karena itu, frekuensi yang sama dapat digunakan kembali di sel terdekat. Maka banyak orang dapat melakukan percakapan secara simultan dalam sel yang berbeda di seluruh kota atau wilayah, meskipun mereka berada dalam satu saluran.

Dalam setiap sel, terdapat stasiun dasar yang berisi antenna *wireless* dan perlengkapan radio lain. Antenna *wireless* dalam setiap sel akan menghubungkan sumber ke jaringan telepon *local*, internet, ataupun jaringan *wireless* lain. Antenna *wireless* mentransimiskan sinyal.

2.5 Jaringan *Backbone*

Jaringan *backbone* adalah saluran atau koneksi berkecepatan tinggi yang menjadi lintasan utama dalam sebuah jaringan. *Network Backbone* adalah *network* yang menghubungkan beberapa jaringan dengan berkecepatan rendah melalui *gateway*. Jaringan ini tentunya harus memiliki *bandwidth* yang lebar, kecepatan transmisi yang tinggi dan dibangun dengan infrastruktur yang memiliki keandalan yang tinggi. Jaringan ini memiliki kapasitas *bandwidth* yang besar (10 Gbps), titik transit dari pertemuan banyak jalur, dan menggunakan media akses *Wireline Fiber Optic*/*Wireless BWA*.

Teknologi *backbone* yang sering digunakan, yaitu teknologi satelit, *microwave,* dan *fiber optic*. Masing-masing mempunyai kelebihan dan kekurangan. Yang mudah diimplementasikan adalah menggunakan teknologi satelit tetapi *delay* satelit sebesar 500ms cukup membuat percakapan telepon menjadi tidak nyaman, untuk keperluan data sebelumnya tidak terlalu bermasalah. Belakangan ini mulai muncul jasa komunikasi data menggunakan satelit pada Ku-Band yang di kenal dengan DVB-RCS. Jasa ini agak murah dengan konsekuensi biasanya akan putus sekitar 5 menit jika ada awan hujan diatas antenna.

Teknologi *microwave* biasanya digunakan untuk jarak yang tidak terlalu jauh, umumnya setiap 40-50km di pasang *relay* untuk membangun jaringan *microwave* jarak jauh. Teknologi *microwave* seperti Wi-Fi dan WiMAX menjadi primadona bagi pengguna jaringan data kecepatan tinggi sampai dengan 100Mbps terutama untuk mencapai

tujuan-tujuan di dalam kota. Biaya menjadi salah satu faktor kesuksesan implementasi WiFi & WiMAX di sebuah kota. Teknologi telepon 3G mulai di dengungkan belakangan ini; tapi dengan kecepatan 3G yang hanya sekitar 2Mbps, sebetulnya tidak menarik bagi backbone data jarak jauh.

Untuk sambungan data kecepatan tinggi jarak jauh, pilihan media komunikasi yang paling baik adalah fiber optik. Fiber optik mempunyai kecepatan sangat tinggi dan delay sangat rendah. Pada hari ini, tidak banyak operator telekomunikasi di Indonesia yang berani mengimplementasi backbone fiber optik jarak jauh, setahu saya, hanya Telkom, PLN, XL dan sedikit Indosat.

Alasan penggunaan jaringan *backbone,* antara lain :

1. Semakin meningkatnya kebutuhan interkoneksi antar jaringan lokal yang ada.
2. Meningkatnya kecepatan *transfer* data khususnya untuk data grafis, video, dan audio, karena kecepatan transfer data FDDI dapat mencapai 100 Mbps.
3. Konsep instalasi dan manajemen jaringan *backbone* lebih sederhana, tetapi jarak jangkauan dapat lebih luas dan jauh.
4. Jaringan *backbone* dapat meningkatkan kemampuan dan mengatasi *bottleneck transfer*.

Desain jaringan Utama (*Backbone*) :

a. Teknologi dalam membangun jaringan *backbone*
   - *Bridge backbone ring*
   - *Fiber Distributed Data Interface* (FDDI), 100 Mbps, sistem dual *ring* dengan protokol MAC token *ring*
     FDDI merupakan protokol yang digunakan untuk transmisi pada jaringan yang mempunyai *Token Passing Ring* yang dapat meningkatkan kinerja jaringan. FDDI menggunakan serat optik dengan kecepatan transmisi mencapai 100 Mpbs. FDDI dapat menghubungkan sampai 500 terminal dengan jarak maksimum 2 km.
   - *Asynchronous Transfer Mode* (ATM), lokal *switch, public switch*
     Asynchronous Transfer Mode (ATM) merupakan teknologi ini dikembangkan pada awal tahun 1990-an. Prinsip pada ATM adalah setiap informasi harus ditransfer ke dalam bentuk sel. ATM memiliki kecepatan transfer data yang tinggi, yaitu mencapai 150 Mpbs. Teknologi ini sangat cocok digunakan dalam pengiriman data dalam bentuk suara atau gambar (multimedia).

b. Teknik pengkabelan

Sistem kabel pada jaringan *backbone* harus menyediakan interkoneksi antara ruang peralatan komunikasi, ruang telekomunikasi, ruang terminal utama, dan fasilitas masuk dalam struktur sistem telekomunikasi kabel. sistem pengkabelan terdiri dari kabel *backbone*, kabel patch atau jumper yang digunakan untuk menghubungkan lalu lintas transfer data. Kabel *backbone* menghubungkan lalu lintas utama data. Warna sebutan untuk tipe kabel serat antara lain:

Single Mode fiber> Kuning
Multi Mode fiber 62.5 micron > *Orange*
Multi Mode fiber 50 micron 1GB > *Orange*
Multi Mode fiber 50 micron 10GB > *Aqua*

c. Topologi jaringan *backbone*

Topologi bus sering juga disebut sebagai topologi *backbone*, dimana ada sebuah kabel coaxial yang dibentang kemudian beberapa komputer dihubungkan pada kabel tersebut.

Secara sederhana pada topologi bus, satu kabel media transmisi dibentang dari ujung ke ujung, kemudian kedua ujung ditutup dengan "terminator" atau *terminating-resistance* (biasanya berupa tahanan listrik sekitar 60 ohm). Pada titik tertentu diadakan sambungan (tap) untuk setiap terminal. Wujud dari tap ini bisa berupa "kabel *transceiver*" bila digunakan "*thick coax*" sebagai media transmisi atau berupa "BNC T-*connector*" bila digunakan "*thin coax*" sebagai media transmisi atau berupa konektor "RJ-45" dan "hub" bila digunakan kabel UTP.

Transmisi data dalam kabel bersifat "*full duplex*", dan sifatnya "*broadcast*", semua terminal bisa menerima transmisi data. Suatu protokol akan mengatur transmisi dan penerimaan data, yaitu Protokol Ethernet atau CSMA/CD. Pemakaian kabel coax (10Base5 dan 10Base2) telah distandarisasi dalam IEEE 802.3

Hirarki Infrastruktur *backbone* internet :

1. *Backbone* Internasional, yaitu jaringan yang menghubungkan trafik domestik ke jaringan internasional, sarana utama yang digunakan adalah kabel optik bawah laut dan satelit internasional dan regional.
2. *Backbone* Domestik, adalah sarana infrastruktur yang menghubungkan kota-kota di seluruh Indonsia. Untuk hubungan kota-kota metropolitan di bagian barat Indonesia pada umumnya telah terhubung dengan kabel optik, sedangkan satelit dan radio teresterial telah terhubung ke seluruh wilayah Indonesia, rincian detail masing-masing bakbone akan diuraikan pada slide selanjutnya.

3. Jaringan Akses, adalah jaringan yang terhubung langsung ke pelanggan. Jaringan inilah yang memerlukan investasi yang sangat besar dengan berbagai macam variasi teknologi. Secara garis besarnya dibagi mencadi dua, yautu yang menggunakan media kabel dan wireless. Umumnya jumlah pelanggan dari masing-masing operator menggambarkan jumlah akses yang tersedia.

2.6 Borland Delphi 7

Delphi adalah sebuah bahasa pemrograman yang bersifat OOP (*Object Oriented Programming*), artinya sebuah program yang mempunyai objek-objek tertentu dalam pemrogramannya. OOP ini memiliki beberapa unsur yaitu : *Encapsulation* (pemodelan), *Inheritance* (penurunan), *Polymorphism* (polimorfisme). Selain itu, Delphi adalah sebuah program yang bersifat visual, artinya mempunyai tampilan grafik-grafik yang mudah dimerngeti oleh pemula sekalipun (*Graphical User Interface*).

Umumnya delphi hanya digunakan untuk pengembangan aplikasi desktop, *enterprise* berbasis *database* dan program-program kecil. Namun karena pengembangan delphi yang semakin pesat dan bersifat *general purpose* bahasa pemrograman ini mampu digunakan untuk berbagai jenis pengembangan *software*. Delphi juga disebut sebagai pelopor perkembangan *RedTool* (*Rapid Application Development*) tahun 1995 sehingga banyak orang yang mulai mengenal dan menyukai bahasa pemrograman yang bersifat VCL (*Visual Component Library*) ini.

Aspek penting yang perlu dicatat tentang Bahasa *Delphi* yaitu :

- Penanganan *object* sebagai *reference/pointer* secara transparan
- Properti sebagai bagian dari bahasa tersebut; benar, sebagai *getter* dan *setter* (atau *accessor and mutator*), yang secara transparan mengenkapsulasi akses pada *field-field* anggota dalam kelas tersebut
- *Prtoperty index* dan *Default* yang menyediakan akses pada data kolektif[4]

2.6.1 *Internet Direct* (Indy)

Indy adalah komponen *open* source yang digunakan untuk membuat aplikasi yang terdiri dari protokol jaringan, seperti HTTP, FTP, SMTP, TCP, UDP, Gopher, Whois, dan lain-lain. Karena *open source* maka semua kode program yang ada dapat dikembangkan kembali.

Komponen *internet direct* (indy) digunakan untuk aplikasi jaringan berbasis *client-server*. Aplikasi jaringan yang dibuat dengan komponen indy terhubung dalam sebuah jaringan *Local Area Network* (LAN), komponen indy terbagi menjadi dua grup utama yaitu indy *server* dan indy *client* dimana indy *server* ditempatkan pada aplikasi *server* sedangkan indy *client* ditempatkan pada aplikasi *client.*

2.7 Quality of Service (QoS)

Quality of Service (QoS) didefinisikan sebagai suatu pengukuran tentang seberapa baik jaringan dan merupakan suatu usaha untuk mendefinisikan karakteristik dan sifat dari suatu layanan. Pada jaringan berbasis IP, IP QoS mengacu pada performansi dari paket-paket IP yang lewat melalui satu atau lebih jaringan.

Kualitas layanan diukur dengan standar dari *Telecommunication and Internet Protocol Harmonization Over Networks* (TIPHON) dan standar dari *International Telecommunication Union-Telecommunication* (ITU-T). Dalam tugas akhir ini parameter QoS yang akan dianalisis adalah *packet loss* dan *throughput*.

2.7.1 *Packet Loss*

Packet loss merupakan banyaknya paket yang gagal mencapai tempat tujuan. Ketika *packet loss* besar maka dapat diketahui bahwa jaringan sedang sibuk atau terjadi *overload*.[7]

Pengukuran *packet loss* sebagai bahan analisis jaringan pada komunikasi data secara *real time* cukup penting. Trafik komunikasi *real time* yang menggunakan transport protokol UDP tidak menjamin sebuah paket data dapat diterima oleh *node* tujuan dengan baik. Berbeda dengan pengiriman paket data menggunakan protokol TCP yang proses pengiriman datanya melalui prose *three-way-handshaking*.

Dengan demikian perlu dipastikan kualitas sebuah jaringan untuk komunikasi data *real time* yang disebut sebagai QoS. Untuk menghitung *packet loss* (dalam persen) digunakan rumus berikut :

$$Packet\ loss\ rate = \left(\frac{total\ packet\ loss}{total\ packet\ sent}\right) \quad 100\% \qquad (1)$$

Berikut ini standar untuk hasil penghitung *packet loss* versi TIPHON

Tabel 2-1 Standar *packet loss* versi TIPHON

Packet Loss	Kualitas
0 %	*Perfect*
3%	*Good*
15%	*Medium*
25%	*Poor*

Sumber : TIPHON

Sedangakn menurut versi ITU-T, terdapat tiga kategori penurunan kualitas jaringan berdasarkan standarisasi nilai *packet loss* sebagai berikut :

Tabel 2-2 Standar *packet loss* versi ITU-T

Packet Loss	Kualitas
3%	Baik
15%	Cukup
25%	Buruk

Sumber : ITU-T G.114

2.7.2 *Throughput*

Throughput diartikan sebagai laju data aktual per satuan waktu (*bits per second*). Biasanya *throughput* selalu dikaitkan dengan *bandwidth*. Karena throughput memang bisa disebut sebagai *bandwidth* dalam kondisi yang sebenarnya. *Bandwidth* lebih bersifat tetap sementara *throughput* sifatnya dinamis tergantung *trafik* yang sedang terjadi.

Cara untuk menghitung *throughput* sebagai berikut :

$$t\ roug\ \ put = \frac{jumlah\ data\ yang\ diterima}{waktu\ pengiriman} \quad (2)$$

2.8 XAMPP

XAMPP merupakan perangkat lunak yang mendukung banyak sistem operasi dan merupakan kompilasi dari beberapa program. Fungsinya dalah sebagai *server* yang berdiri sendiri (*localhost)*. Nama XAMPP sendiri merupakan singkatan dari X (empat sistem operasi apapun), Apache, MySQL, PHP, dan Perl. Dengan menginstall XAMPP maka tidak perlu lagi melakukan instalasi dan konfigurasi *web server* Apache, PHP, dan MySQL secara manual. XAMPP akan menginstalasi dan mengonfigurasikannya secara otomatis.

Dalam satu paket XAMPP tersedia :

1. Apache Cgi-Bin
2. FTP
3. Mercury Mail (SMTP)
4. PHP
5. MySQL
6. Perl
7. PHPMyAdmin
8. Webalizer

2.8.1 Apache

Apache berfungsi untuk menampilkan halaman web yang benar, sesuai dengan PHP yang telah dibuat. Apache bersifat *open source*, artinya setiap orang boleh menggunakan, mengambil, dan mengubah kode programmnya.

2.8.2 *Personal Home Page (*PHP)

PHP merupakan bahasa pemrograman yang digunakan untuk membuat web yang bersifat *server-side scripting*. PHP memungkinkan untuk membuat halaman web yang bersifat dinamis, yakni isi informasi *website* berubah-uah dan interaktif dua arah baik dari pemilik maupun pengguna *website*. PHP dapat dijalankan pada berbagai macam OS, seperti Windows, Linux, dan Mac OS.

Sistem manajemen *database* yang sering digunakan bersama adalah MySQL. Namun PHP juga mendukung sistem manajemen *database* Oracle, Microsoft Access, Interbase, d-Base, PostgerSQL, dan lain-lain. PHP juga bersifat *open source*.

2.8.3 MySQL

Database yang digunakan untuk membangun sebuah aplikasi berbasis *client-server* ini yaitu MySQL. SQL sendiri merupakan kepanjangan dari S*tructured Query Language*. SQL adalah sebuah bahasa pemrograman yang digunakan untuk berkomunikasi dengan *server*, dimana SQL sendiri terbagi menjadi beberapa bagian, yaitu :

- ✓ DML (*Data Manipulation Language*) yaitu bahasa untuk memanipulasi data
- ✓ DDL (*Data Definition Language*) yaitu bahasa utnuk mendefinisikan struktur *database*, seperti CREATE, ALTER, DAN DROP

- ✓ *Transact SQL*, bagian ini adalah bahasa pemrograman yang diletakkan pada *database server*. Konten dari bahasa ini adalah untuk menjaga *automacy* sebuah transaksi pertukaran data.

MySQL merupakan sistem manajemen *database* yang bersifat *open source*. MySQL digunakan untuk membuat dan mengelola *database* beserta isinya, seperti menambahkan, mengubah, dan menghapus data. MySQL juga bersifat *relational*, artinya data-datayang dikelola akan diletakkan pada beberapa tabel terpisah, sehingga proses manipulasi data akan menjadi lebih cepat.

2.8.4 PHPMyAdmin

Salah satu perangkat lunak yang digunakan untuk mengelola *database* dalam MySQL adalah PHPMyAdmin. Dengan PHPMyAdmin kita dapat dengan mudah membuat tabel, mengisi data, dan banyak lagi hal lainnya yang tanpa harus hafal perintahnya, namun cukup dengan mengisi tabel-tabel yang telah tersedia.[2]

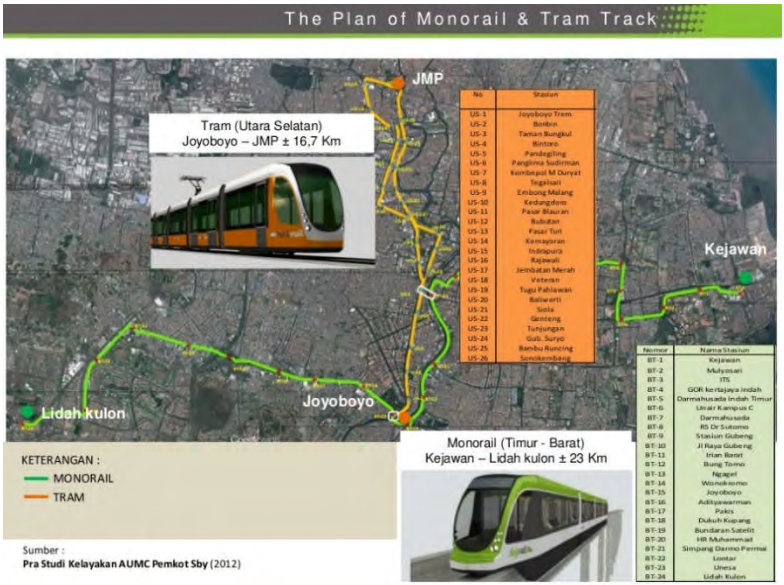
BAB III
PERANCANGAN DAN IMPLEMENTASI SISTEM

Pada bab ini akan membahas tentang proses perancangan dan implemetasi *testbed* komunikasi pengiriman data dari sejumlah titik OBU ke *server* dengan menggunakan Borland Delphi 7. Untuk memudahkan proses perancangan dan implementasi sistem diperlukan alur yang menjelaskan garis besar proses yang dilakukan saat melaksanakan tugas akhir.

3.1 AMC Kota Surabaya

Angkutan Massal Cepat (AMC) di Surabaya memiliki beberapa moda transportasi seperti angkot, minibus, bus, tram, dan monorail. Dalam pengerjaan tugas akhir ini, penulis fokus pada salah dua moda transportasi, yaitu monorail dan tram.

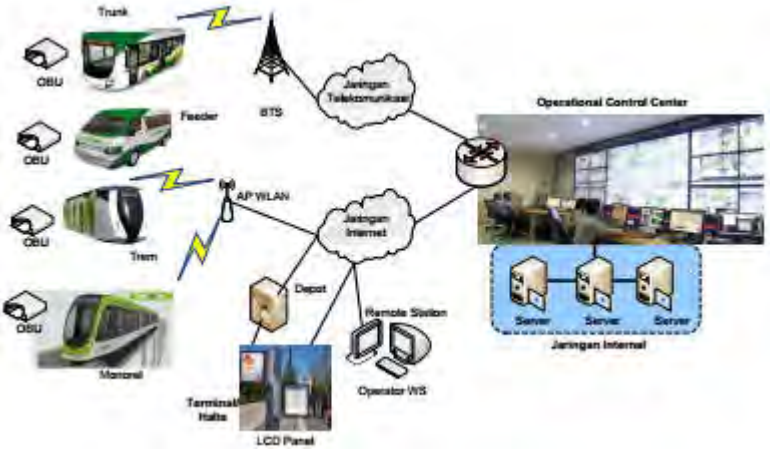
Angkutan massal pada Gambar 3.1 biasa disebut sebagai Suratram dan Boyorail. Kedua jenis moda transportasi ini menghubungkan Surabaya bagian timur-barat dan utara-selatan. Surotram menghubungkan Surabaya bagian utara sampai Surabaya bagian selatan dengan total jarak ± 16,7 Km dan melewati 26 titik halte. Daerah yang dilewati oleh Surotram yaitu Sonokembang, Bambu Runcing, Gub. Suryo, Tunjungan, Genteng, Siola, Baliwerti, Tugu Pahlawan, Veteran, Jembatan Merah, Rajawali, Indrapura, Kemayoran, Pasar Turi, Bubutan, Pasar Blauran, kedungdoro, Embong Malang, Tegalsari, Kombepol M Duryat, Panglima Sudirman, Pandegiling, Bintoro, Taman Bungkul, Bonbin, dan Joyoboyo. Boyorail menghubungkan Surabaya bagian timur sampai bagian barat dengan total jarak ± 23 Km dan melewati 24 titik halte. Daerah yang dilewati oleh Boyorail yaitu Kejawan, Mulyosari ITS, Gor Kertajaya, Dharmahusada Indah Timur, RS DR Sutomo, Stasiun Gubeng, Jl. Raya Gubeng, Irian Barat, Bung Tomo, Ngagel, Wonokromo, Joyoboyo, Adityawarman, Pakis, Dukuh Kupang, Bundaran Satelit, HR Muhammad, Simpan Darmo Permai, Lontar, Unesa, dan Lidah Kulon. Berikut gambar yang mengenai jalur monorail dan tram:



Gambar 3.1 Jalur Monorail dan Tram

3.1.1 Sistem Jaringan Komunikasi AMC

Berangkat dari desain manajemen armada kota Surabaya dapat dibagi dan didefinisikan dari masing-masing bagian yang menjadi pekerjaan yang disusun sebagai tahap-tahap penerapan manajemen armada kota Surabaya. Berikut desain majanemen armada kota Surabaya:



Gambar 3.2 Desain Manajemen Armada AMC

Adapun tahap-tahap penerapan manajemen armada kota Surabaya sebagai berikut :

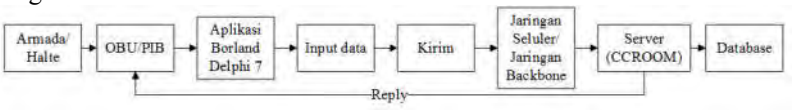
1. *Phase* 1 CCROOM-*Feeder Monitoring*
   Pembangunan CCROOM yang disiapkan untuk mampu memonitoring *feeder vehicle* dan halte. Pembangunan ini membutuhkan pengadaan jaringan *hardware* dan *software* untuk internal CCROOM untuk meningkatkan kemampuan yang sekarang sudah ada.
2. *Phase 2* OBU *Implementation*
   Pada *phase* ini diawali dengan mengadakan OBU yang mempunyai modul GSM/GPS, *central unit,* piranti *human interface*, *diagnostic adapter*, modul I/O, unit *power supply* dan baterai yang kemudian diimplementasi pada masing-masing unit armada dan dilakukan uji coba monitoring.
3. *Phase* 3 *Fleet Optimation*
   Melakukan optimasi armada dengan menggunakan *software* yang mampu memberikan rekomendasi *routing* yang efisien dan berskala bisnis sehingga bisa memberikan penghematan bahan bakar, mengurangi waktu tempuh, dan waktu kerja sehingga dapat meningkatkan *revenue* dan mengurangi biaya operasi

4. *Phase* 4 *Operation and Maintenance*
   Phase ini untuk memberi jaminan bahwa sistem informasi dapat berfungsi sesuai dengan yang direncanakan dan memberikan unjuk kerja yang optimal sampai sistem habis masa penggunaannya.

Tugas akhir ini merupakan salah satu bagian dari *phase* 1. Dalam tugas akhir ini akan dibuat suatu aplikasi yang digunakan untuk pengujian pengiriman data dari OBU/PIB ke CCROOM. Pengujian pengiriman data ini bertujuan untuk mengetahui kualitas sistem dan mendapatkan rekomendasi panjang data yang akan digunakan pada kondisi tertentu.

3.1.2 Struktur Sistem Jaringan Komunikasi AMC

Sistem jaringan komunikasi AMC dapat dilihat pada blok diagram berikut :

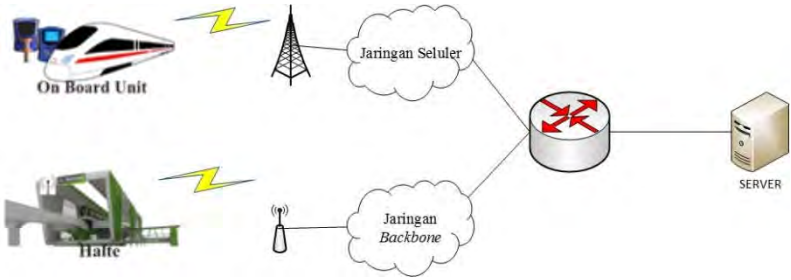


Gambar 3.3 Diagram blok sistem jaringan komunikasi AMC

On-Board Unit (OBU) berada pada tiap armada sedangkan *Passenger Information Board* (PIB) berada pada halte. Pada OBU dan PIB akan di *install* aplikasi yang telah dibuat menggunakan *software* Borland Delphi 7. Untuk mengirimkan data ke *server* (CCROOM), *client* harus memasukkan beberapa data sesuai dengan kolom yang terdapat pada aplikasi. Kemudian data dikirim ke *server.* Jika pengiriman data menggunakan OBU, maka data dikirim melalui jaringan seluler. Jika pengiriman data menggunakan PIB, maka data dikirim melalui jaringan *backbone*. Setelah data diterima oleh *server, server* akan memberi balasan ke *client.* Selanjutnya data dimasukkan ke *database* untuk diolah.

3.2 Alur Perancangan Sistem Komunikasi Terintegrasi AMC

Bagian ini merupakan bagian awal dari sistem yang dirancang, maka dari itu perlu dilakukan tahapan perencanaan dan implementasi sistem. Perencanaan dibuat untuk memudahkan pekerjaan dari implementasi sistem yang akan dirancang, sehingga analisis terhadap implementasi dapat dipahami. Maka dibuatlah sebuah diagram alur.



Gambar 3.4 Diagram alur komuikasi terintegrasi AMC

Gambar 3.4 menjelaskan berlangsungnya komunikasi dari *On-Board Unit* (OBU) dan *Passenger Information Board* (PIB) sampai dengan *server*. Komunikasi tersebut akan mengirimkan beberapa data ke *server*, antara lain:

1. *E-ticketing* berupa ID
2. Lokasi *client* berupa *longitude* dan *latitude*
3. Waktu pengiriman berupa tanggal, *time send,* dan *time receive*
4. Jumlah paket yang dikirim
5. IP *client*

3.2.1 Proses Komunikasi OBU dan PIB ke Jaringan

Dalam manajemen cerdas ini akan dipasang *On-Board* Unit (OBU) pada masing-masing armada dan *Passenger Information Board* (PIB) pada setiap halte. Untuk dapat terhubung ke jaringan, OBU terkoneksi ke salah satu jaringan operator sedangkan PIB terkoneksi ke *access point.*

3.2.2 Proses Komunikasi Jaringan ke *Server*

Komunikasi berikutnya adalah komunikasi dari jaringan ke *server*. Pada komunikasi ini, data yang tersimpan sementara di OBU dan PIB akan dikirim ke *server*. Proses pengiriman data dari OBU melalui jaringan seluler, karena OBU dipasang di dalam armada yang mana akan selalu berpindah-pindah tempat. Sedangkan pengiriman data dari PIB melalui jaringan *backbone*, karena PIB akan dipasang pada halte yang terdapat *access point* disana.

3.3 Perangkat Server yang Digunakan

Server yang digunakan dalam tugas akhir ini adalah its2.lawanghosting.pw. *Server* its2.lawanghosting.pw merupakan *server*

untuk publikasi alamat IP-*Public speedy*. *Server* ini dianalogikan sebagai *server* yang terdapat pada CC-ROOM. *Server* ini dirancang untuk menguji pengiriman dan penerimaan data dari *On-Board Unit* (OBU) dan *Passenger Information Board* (PIB) ke CC-ROOM.

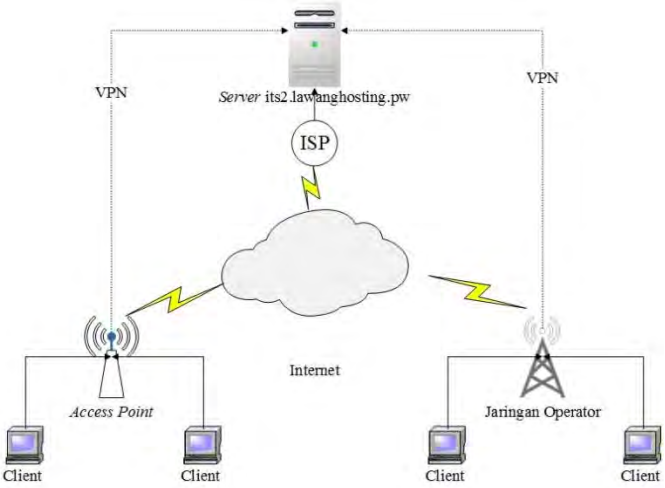
3.4 Perancangan Simulasi Sistem Komunikasi Terintegrasi AMC

Pada bagian ini akan dilakukan proses perancangan *testbed* sistem komunikasi terintegrasi untuk AMC. Perancangan dilakukan dengan membuat simulasi sistem *client-server* menggunakan Borland Delphi 7. Adapun tahapan-tahapan perancangan simulasi :

1. Pembuatan arsitektur jaringan pengiriman data
2. Perancangan sistem *client-server* untuk pengiriman data pada Borland Delphi 7
3. Perancangan *Database* PHPMyAdmin untuk penyimpanan data.

3.4.1 Pembuatan Arsitektur Jaringan Pengiriman Data

Pada tahap ini akan dilakukan pemodelan arsitektur jaringan pengiriman data dari *client* ke *server*. Arsitektur jaringan ini digunakan untuk memudahkan saat melakukan *testbed* nanti. Berikut arsitektur jaringan pengiriman data :

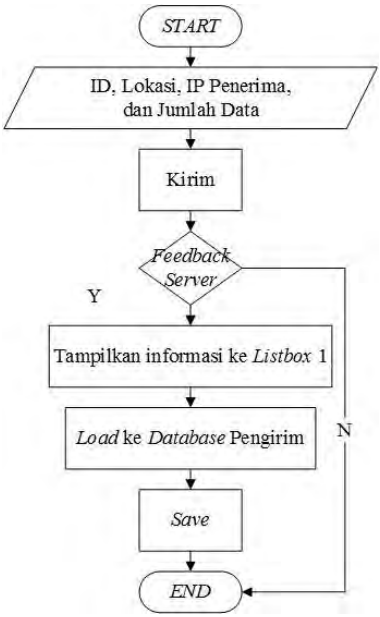


Gambar 3.5 Arsitektur jaringan pengiriman data

Arsitektur dalam Gambar 3.5 merupakan arsitektur yang digunakan untuk melakukan *testbed* komunikasi sistem terintegrasi AMC. Dalam gambar tersebut terdapat beberapa *client* sebagai PIB pada halte yang terhubung dengan *access point* yang dianalogikan sebagai jaringan *backbone*. Juga terdapat beberapa *client* sebagai OBU pada armada yang terhubung dengan jaringan operator seluler. Kemudian data dikirim ke *server* yang memiliki IP *Public*. *Testbed* ini dilakukan dalam jaringan *Virtual Private Network* (VPN).

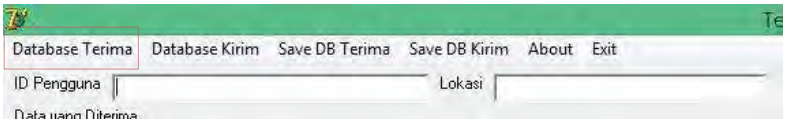
3.4.2 Perancangan Aplikasi untuk Pengiriman Data pada Borland Delphi 7

Aplikasi ini dibuat untuk mensimulasikan proses pengiriman data dari *client* ke *server* agar dapat melakukan *testbed* komunikasi terintegrasi AMC. Aplikasi yang dibuat telah dirancang berfungsi sebagai *client*. Sebelum dibuat aplikasinya, perlu dibuat suatu alur atau *flowchart* aplikasi pengiriman data pada Borland Delphi 7 untuk *testbed*. Berikut *flowchart* aplikasi *client* Borland Delphi 7 :



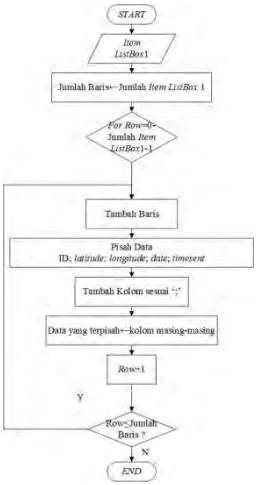
Gambar 3.6 *Flowchart* aplikasi untuk uji coba

Client koneksi ke jaringan internet. *Client* memasukkan atau meng-*input*kan data ID, Lokasi, IP penerima, dan jumlah data pada aplikasi Borland Delphi 7 yang telah dibuat. Kemudian data-data tersebut dikirim ke *server*. *Server* telah terhubung dengan *database* PHPMyAdmin. *Client* akan mendapat *feedback (umpan balik)* setelah data terkirim ke *server*. *Feedback* tersebut berupa informasi yang muncul pada *ListBox* 1. Informasi yang didapat yaitu balasan dari *server* berupa ID, lokasi, tanggal pengiriman, dan waktu pengiriman per paket data. Balasan dari *server* dapat di konversi ke *database* dengan format CSV (*Comma Separated Value*). Caranya adalah dengan menekan "*Database* Pengirim", maka balasan dari *server* tersebut akan di muat terlebih dahulu dalam bentuk tabel yang ditampilkan pada *StringGrid* 1.



Gambar 3.7 *Toolbar* untuk *Database* Terima

Adapun penjelasan proses *load* data ke *StringGrid*1 dalam bentuk diagram alur pada Gambar 3.8 :

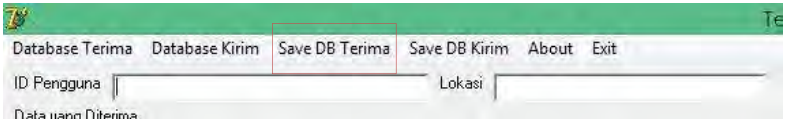


Gambar 3.8 *Flowchart* proses *load* ke dalam bentuk tabel

Pertama, data dari *ListBox* 1 diolah untuk ditampilkan dalam bentuk tabel pada *StringGrid* 1. Jumlah baris tabel *StringGrid*1 sebanyak jumlah *item ListBox*1 (data masuk). Pemisahan masukan data

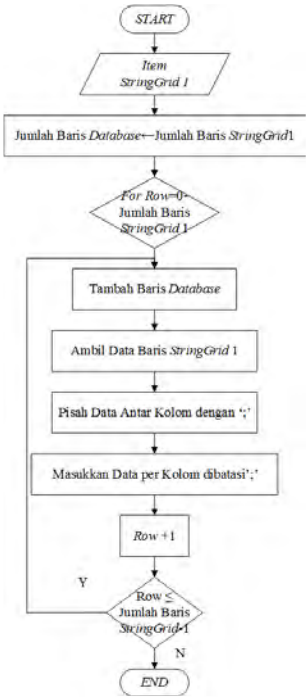
per kolom menggunakan ";" sebagai acuan. Tabel pada aplikasi ini terdapat 5 kolom, yaitu *Identity*, *latitude*, *longitude*, *date*, dan *Time*Sent. Jika data yang masuk sudah sesuai dengan baris dan kolom yang diinginkan, proses berakhir.

Kemudian tabel dapat disimpan ke memori komputer dengan menekan "*Save* DB Pengirim".



Gambar 3.9 *Toolbar* untuk *Save Database*

Alur pemrograman untuk menyimpan data ke format CSV dijelaskan pada Gambar 3.10 :



Gambar 3.10 *Flowchart* proses *save* dalam format CSV

Data *input* merupakan *item* dari *StringGrid*1. Jumlah baris pada *StringGrid*1 dijadikan acuan jumlah baris *file* CSV. Proses selanjutnya data pada setiap baris *StringGrid*1 akan dikonversi menurut aturan CSV. Pemisah data antar kolom (*delimiter*) menggunakan ";". Jumlah kolom pada tabel ada lima. Sehingga pada *file* CSV terdapat empat ";" untuk memisahkan data antar kolom. Pemisahan data pada *file* CSV menjadi "*Identity*, *latitude*, *longitude*, *date, Time*Sent". Jika sudah sesuai data dari *StringGrid*1 maka proses berakhir.

Setelah pembuatan alur aplikasi, dibuatlah aplikasi pengiriman dan penerimaan data pada Borland Delphi 7. Dibutuhkan beberapa komponen *form,* yaitu:

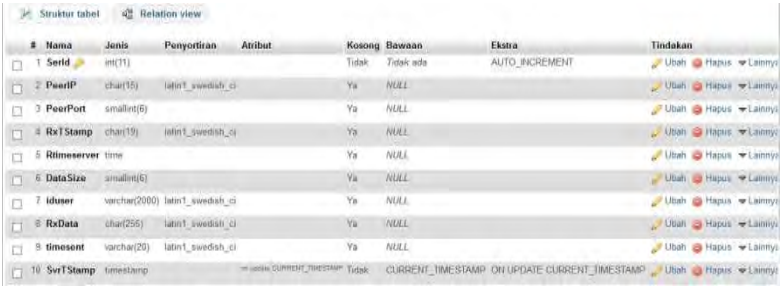
1. UDP, berfungsi untuk mengirim dan menerima data.
2. *Save Dialog,* berfungsi untuk menampilkan jendela *save database*.
3. *Timer,* berfungsi untuk membangkitkan waktu pada aplikasi.
4. *Menu files*, berfungsi untuk membuat desain menu pada aplikasi.
5. *StringGrid*, berfungsi untuk membuat tabel pada aplikasi
6. *ListBox,*berfungsi untuk menampilkam informasi yang dibutuhkan.
7. *Button,* berfungsi sebagai komponen tambahan yang membutuhkan tombol.
8. *Edit text*, berfungsi sebagai komponen tambahan untuk memasukkan atau mengeluarkan sebuah informasi.
9. *Label*, berfungsi sebagai pemberi keterangan yang dibutuhkan.
10. *Memo,* berfungsi untuk menampilkan informasi.

Tampilan (*interface*) aplikasi yang dibuat pada Borland Delphi 7 dapat dilihat pada Gambar 3.11 :

Gambar 3.11 Tampilan aplikasi untuk uji coba

3.4.3 Perancangan *Database* PHPMyAdmin untuk Penyimpanan Data

Server yang digunakan untuk menerima data dalam tugas akhir ini adalah *server* its2.lawanghosting.pw dengan memanfaatkan *database* PHPMyAdmin pada *server*. Data yang masuk ke *server* akan disimpan dalam *database* its2016a. *Database* dibuat sesuai dengan format perancangan sistem komunikasi terintegrasi AMC. Tabel yang terdapat pada *database* yaitu tabel udp_rxlog. Berikut ini Gambar 3.12 yang merupakan tampilan struktur dari tabel udp_rxlog :


Struktur tabel | Relation view

#	Nama	Jenis	Penyortiran	Atribut	Kosong	Bawaan	Ekstra	Tindakan
1	SerId	int(11)			Tidak	Tidak ada	AUTO_INCREMENT	Ubah Hapus Lainnya
2	PeerIP	char(15)	latin1_swedish_ci		Ya	NULL		Ubah Hapus Lainnya
3	PeerPort	smallint(6)			Ya	NULL		Ubah Hapus Lainnya
4	RxTStamp	char(19)	latin1_swedish_ci		Ya	NULL		Ubah Hapus Lainnya
5	Rtimeserver	time			Ya	NULL		Ubah Hapus Lainnya
6	DataSize	smallint(6)			Ya	NULL		Ubah Hapus Lainnya
7	iduser	varchar(2000)	latin1_swedish_ci		Ya	NULL		Ubah Hapus Lainnya
8	RxData	char(255)	latin1_swedish_ci		Ya	NULL		Ubah Hapus Lainnya
9	timesent	varchar(20)	latin1_swedish_ci		Ya	NULL		Ubah Hapus Lainnya
10	SvrTStamp	timestamp		on update CURRENT_TIMESTAMP	Tidak	CURRENT_TIMESTAMP	ON UPDATE CURRENT_TIMESTAMP	Ubah Hapus Lainnya

Gambar 3.12 Tampilan struktur dari tabel udp_rxlog

Gambar 3.12 adalah tampilan tabel udp_rxlog. Dalam tabel tersebut terdapat beberapa *field* antara lain :

1. SerID, berisi id ID urutan data yang masuk ke *server*
2. PeerIP, berisi IP *client* yang mengirimkan data.
3. PeerPort, berisi port berapa yang dilewati
4. RxTStamp, berisi waktu data diterima *database server*.
5. Rtimeserver, berisi waktu data diterima oleh *server*.
6. Datasize,berisi ukuran *byte* data yang diterima
7. RxData, berisi data yang diterima. Data-data yang diterima adalah ID *client, longitude,* dan *latitude*
8. Timesent, berisi waktu pengiriman oleh *client*
9. SvrTStamp, berisi waktu data masuk ke *database server*

3.5 Implementasi Sistem Komunikasi Terintegrasi AMC

Pada pengerjaan tugas akhir ini, sistem yang telah dibuat akan diimplementasikan. Implementasi sistem yang pertama yaitu implementasi sistem *client-server.* Sistem dibuat dengan membuat aplikasi menggunakan Borland Delphi 7 yang digunakan oleh *client*

untuk mengirim data yang akan diterima oleh *server*. Implementasi yang kedua adalah implementasi dengan jaringan. Dalam tugas akhir ini, sistem akan diimplementasikan dengan dua jenis jaringan, yaitu jaringan seluler dan jaringan *backbone*.

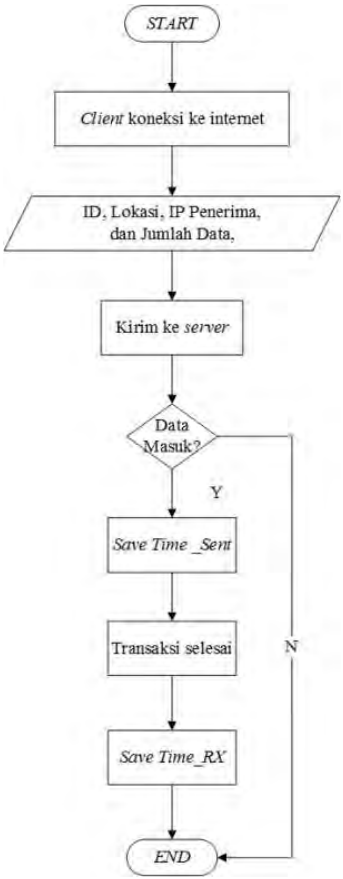
3.5.1 Implementasi Sistem *Client-Server*

Dalam pengerjaan tugas akhir ini, implementasi sistem dilakukan menggunakan *server* its2.lawanghosting.pw dengan memanfaatkan *database* its2016a yang terdapat pada PHPMyAdmin. Aplikasi yang digunakan oleh *client* untuk mengirim data telah dibuat dengan menggunakan *software* Borland Delphi 7.

Pertama kali yang dilakukan adalah *client* harus koneksi ke internet. Selanjutnya, *client* memasukkan data-data yang akan dikirim ke *server*. Data-data tersebut adalah ID, Lokasi, IP penerima, dan jumlah data. Data ID digunakan untuk mengetahui nomor identitas pengguna sistem *e-ticketing*. Data lokasi terdiri dari *latitude* dan *longitude*. *Latitude* merupakan lokasi *client* berdasar koordinat garis lintang yang mengarah dari khatulistiwa (0) ke kutub selatan (sudut 0 ke 90) atau khatulistiwa ke kutub utara (0 ke -90), sedangkan *longitude* merupakan lokasi *client* berdasar koordinat garis bujur seperti khatulistiwa (0 ke 180 atau 0 ke -180). Data IP penerima digunakan untuk mengetahui IP tujuan pengiriman data. Memasukkan jumlah data pada aplikasi *testbed* komunikasi terintegrasi AMC digunakan untuk mengirim sejumlah data sesuai yang diinginkan.

Setelah memasukkan data, data dikirim ke *server*. Kemudian *server* menyimpan waktu pengiriman (*Timesent*) per paket data yang diterima. Proses ini berlangsung hingga paket-paket yang dikirim oleh *client* diterima oleh *server*. Setelah transaksi selesai *server* menyimpan waktu penerimaan (*RTimeserver*).

Berikut Gambar 3.13 implementasi sistem *client-server* yang dijelaskan dalam sebuah alur :

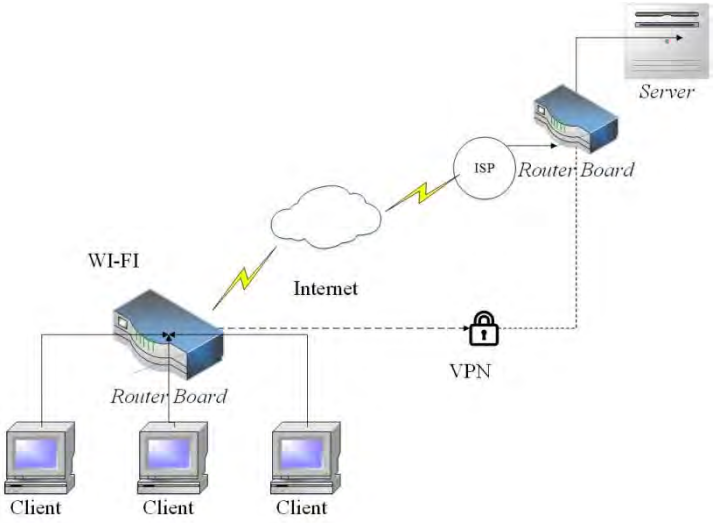


Gambar 3.13 *Flowchart* implementasi sistem *client-server*

3.5.2 Implementasi Jaringan

Pada perancangan tugas akhir ini akan diimplementasikan aplikasi yang telah dibuat untuk mengirimkan data ke *server* melalui jaringan sesuai dengan rancangan sebelumnya. Implementasi ini dilakukan pada dua jenis jaringan. Jenis pertama adalah jaringan *backbone* dengan menggunakan jaringan *Wireless Fidelity* (WI-FI). Jenis yang kedua adalah jaringan seluler dengan menggunakan jaringan operator seluler.

Pengimplementasian aplikasi pengiriman data dari *client* ke *server* yang pertama adalah melalui jaringan *backbone* yang bertujuan untuk menguji coba pengiriman data dari PIB ke CC-ROOM. Pertama kali yang dilakukan adalah membuka aplikasi *testbed* komunikasi terintegrasi yang telah dibuat dengan menggunakan aplikasi Borland Delphi 7 pada tiap laptop yang akan digunakan. Laptop yang digunakan berjumlah dua buah. Laptop-laptop tersebut sebagai *client* yang akan mengirim data ke *server* its2.lawanghosting.pw yang memiliki IP *Public*. Berikut ini Gambar 3.14 mengenai implementasi jaringan *backbone* :

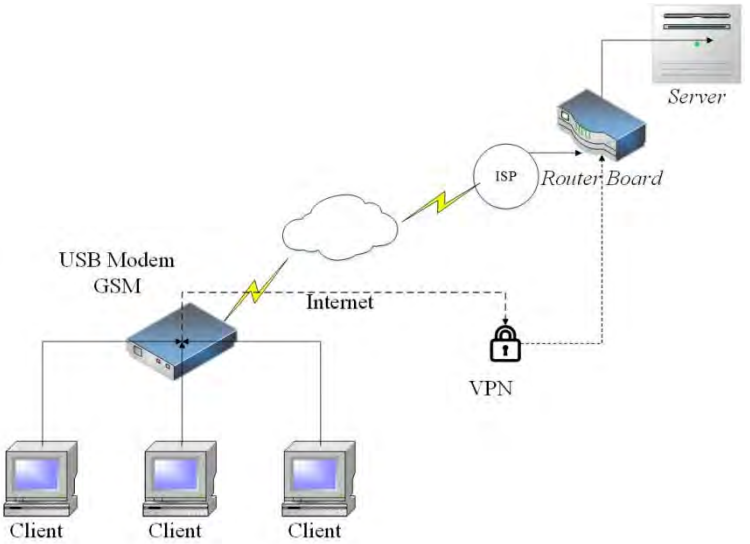


Gambar 3.14 Implementasi jaringan *backbone*

Client dapat terhubung ke *server* melalui jaringan *Virtual Private Network* (VPN). VPN merupakan suatu koneksi antara satu jaringan dengan jaringan lain secara *private* melalui jaringan internet (publik). VPN menggunakan jaringan internet sebagai media perantaranya alias koneksinya bukan secara langsung. Untuk dapat terhubung dengan internet, *client* harus terkoneksi dengan jaringan publik. Jaringan publik yang digunakan adalah jaringan wifi.

Pengimplementasian aplikasi pengiriman data dari *client* ke *server* yang kedua adalah melalui jaringan seluler yang bertujuan untuk menguji coba pengiriman data dari OBU ke CC-ROOM. Langkah-

langkah yang dilakukan dalam pengimplementasian jaringan seluler ini hampir sama dengan langkah-langkah pada pengimplementasian jaringan *backbone*. Pada pengimplementasian ini *client* terkoneksi dengan jaringan operator seluler sebagai jaringan publiknya. Berikut Gambar 3.15 mengenai implementasi jaringan seluler :



Gambar 3.15 Implementasi jaringan seluler

3.6 Skenario Pengujian Sistem Komunikasi Terintegrasi AMC

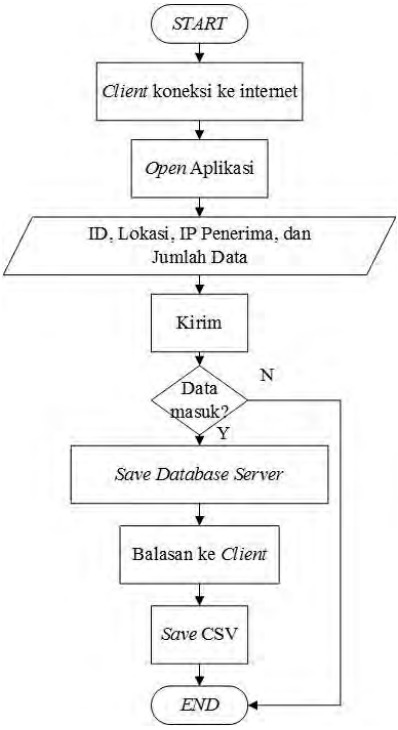
Setelah proses implementasi selesai, langkah selanjutnya yaitu menentukan skenario yang akan digunakan untuk pengujian sistem yang telah dirancang. Skenario tersebut meliputi skenario pengujian sistem *client-server* dan skenario pengujian jaringan. Pada pengujian jaringan, dibuat skenario pengujian QoS yang meliputi *packet loss* dan *throughput.*

3.6.1 Skenario Pengujian Sistem *Client-Server*

Pada bagian ini akan dijabarkan mengenai skenario pengujian sistem *client-server* komunikasi terintegrasi Angkutan Masal Cepat (AMC) yang telah dibuat. Pengujian ini bertujuan untuk mengetahui

apakah sistem telah sesuai dengan rancangan atau belum. Pengujian ini dilakukan dengan cara melakukan proses pengiriman data dari *client* ke *server* menggunakan aplikasi Borland Delphi 7 yang telah dibuat untuk *testbed* komunikasi terintegrasi AMC. Proses ini dilakukan untuk mengetahui apakah data yang dikirimkan dapat diterima oleh *server* dan *client* mendapat balasan dari *server*.

Selain itu, pengujian ini juga dilakukan untuk mengetahui apakah data yang diterima oleh *server* dapat disimpan dalam *database server* udp_rxlog, begitu juga balasan yang diterima oleh *client* apakah dapat disimpan dalam bentuk CSV atau tidak.



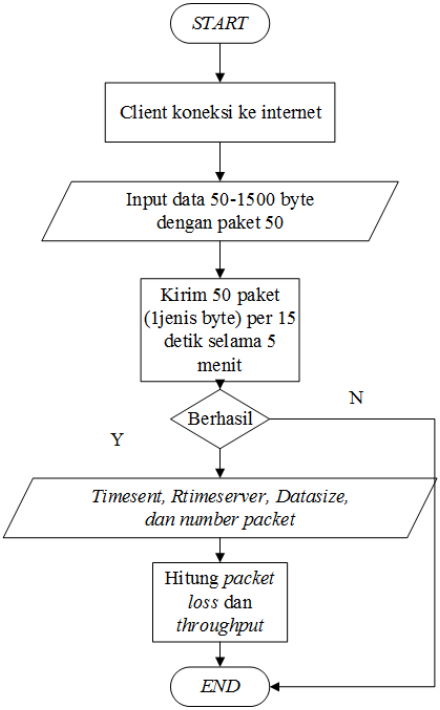
Gambar 3.16 *Flowchart* pengujian *client-server*

Alur pengujian *client-server* ini seperti digambarkan pada Gambar 3.16.Proses pengiriman data diawali dengan *client* koneksi ke jaringan internet yang terdapat pada *access point* (jaringan *backbone*) atau jaringan seluler. Kemudian *client* membuka aplikasi Borland

Delphi 7 untuk *testbed* komunikasi terintegrasi AMC dan mengisi data-data yang terdapat pada aplikasi, yaitu: ID, lokasi, IP penerima, dan jumlah data. Selanjutnya dikirim ke *server*. Apabila data berhasil dikirim ke *server*, data akan disimpan dalam *database server* dan *client* akan mendapat balasan pengiriman. Balasan pengiriman dapat disimpan dalam bentuk CSV. Jika proses pengiriman data dari *client* ke *server* tidak berhasil, maka proses berhenti.

3.6.2 Skenario Pengujian Jaringan

Berikut *flowchart* mengenai skenario pengujian jaringan :



Gambar 3.17 *Flowchart* pengujian jaringan

Pada pengujian ini, akan dilihat QoS jaringan yang telah dirancang. QoS jaringan yang ditinjau adalah *packet loss* dan *throughput.* Langkah-langkah pengujian yang dilakukan yaitu aplikasi telah terkoneksi dengan jaringan Wi-Fi atau jaringan salah satu operator

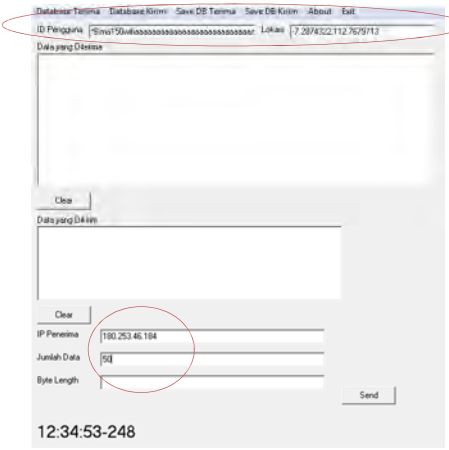
seluler. Kemudian mengatur besar *byte* data sekitar 50-1500 *byte* dan paket yang dikirim sebesar 50 paket yang di-*input*kan oleh *client* pada aplikasi *testbed* komunkasi terintegrasi AMC. 50 paket ini adalah mewakili 50 OBU dan PIB. Pengaturan *byte* data tersebut bertujuan untuk mendapatkan rekomendasi berapa ukuran data terbaik yang dapat diterima oleh *server,* data tersebut dikirim melalui jaringan *backbone* dan jaringan seluler. *Range byte* 50-1500 *byte* tersebut diambil dari standar MTU (*Maximum Transmission Unit*), karena jika ukuran data yang dikirim lebih besar dibandingkan nilai MTU maka paket yang dikirim lebih besar dibandingkan nilai MTU, paket data yang dikirimkan akan terpecah menjadi beberapa fragmen yang akhirnya tidak jadi terkirim dengan benar. Selanjutnya, data dikirim ke *server*. Pengiriman data ke *server* per 50 paket satu jenis *byte* dengan jeda 15 detik sekali kirim dengan rentang waktu 5 menit. Jika berhasil maka didapatkan *output timesent, Rtimeserver, datasize,* dan *number packet* yang disimpan dalam *database.Timesent, Rtimeserver, datasize,* dan *number packet* digunakan untuk menghitung QoS, yaitu *packet loss* dan *throughput*. Apabila data tidak berhasil terkirim ke *server* maka proses diakhiri.

BAB IV
PENGUJIAN DAN ANALISIS

Pada bab ini akan dibahas pengujian dan analisis dari perancangan *testbed* sistem komunikasi terintegrasi untuk AMC dengan menggunakan Borland Delphi 7 dimulai dari proses pengiriman data oleh *client* hingga diterima oleh *server*. Pengujian dibagi menjadi 2 bagian, bagian pertama yaitu pengujian sistem *client-server* komunikasi terintegrasi AMC, bagian kedua yaitu pengujian jaringan komunikasi terintegrasi AMC. Pengujian jaringan komunikasi terintegrasi AMC meliputi pengujian *packet loss* dan *throughput*.

4.1 Pengujian Sistem *Client-Server* Komunikasi Terintegrasi AMC

Sesuai dengan skenario pengujian pada Bab 3, pada pengujian ini akan dilakukan pengujian sistem *client-server*. Pengujian ini bertujuan untuk mengetahui apakah data yang dikirim dapat diterima oleh *server* serta disimpan dalam *database server* dan *client* mendapat balasan dari *server* yang dapat disimpan dalam bentuk CSV. Seperti yang dijelaskan pada bab perancangan sistem, pertama kali yang dilakukan yaitu memasukkan data yang akan dikirim pada aplikasi *testbed.* Data tersebut yaitu ID pengguna, lokasi, IP penerima, dan jumlah data. Tampilan pada Gambar 4.1 :



Gambar 4.1 Tampilan *input* data

Selanjutnya kirim data ke *server*. Jika paket data diterima oleh *server*, maka data disimpan di dalam *database server*. Data dimasukkan sesuai dengan *field-field* yang telah dirancang. Berikut tampilan penyimpanan data dalam *database server* :

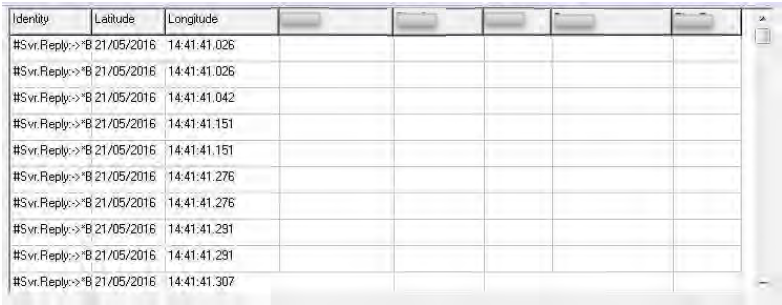
	Serld	PeerIP	PeerPort	RxTStamp	RtImeserver	DataSize	iduser	RxData	timesent	SvrTStamp
Ubah Salin Hapus	1	202.67.40.9	22222	20160519-164431-510	15:44:31	100	NULL	100Tifanyaaaaaaaaa	15:44:30.874	2016-05-20 06:07:29
Ubah Salin Hapus	2	202.67.40.9	22222	20160519-164431-618	15:44:31	100	NULL	100Tifanyaaaaaaaaa	15:44:30.983	2016-05-20 06:07:29
Ubah Salin Hapus	3	202.67.40.9	22222	20160519-164431-592	15:44:31	100	NULL	100Tifanyaaaaaaaaa	15:44:31.092	2016-05-20 06:07:29
Ubah Salin Hapus	4	202.67.40.9	22222	20160519-164432-035	15:44:32	100	NULL	100Tifanyaaaaaaaaa	15:44:31.202	2016-05-20 06:07:29
Ubah Salin Hapus	5	202.67.40.9	22222	20160519-164432-039	15:44:32	100	NULL	100Tifanyaaaaaaaaa	15:44:31.311	2016-05-20 06:07:29
Ubah Salin Hapus	6	202.67.40.9	22222	20160519-164432-102	15:44:32	100	NULL	100Tifanyaaaaaaaaa	15:44:31.420	2016-05-20 06:07:29

Gambar 4.2 Tampilan *database server*

Server mengirim balasan ke *client* bahwa data yang dikirim telah diterima oleh *server*. Balasan dari *server* ditampilkan pada *Listbox1*aplikasi *testbed* komunikasi terintegrasi AMC yang telah dibuat menggunakan *software* Borland Delphi 7. Tampilan balasan dari *server* pada Gambar 4.3 *:*

Gambar 4.3 Balasan dari *server*

Balasan dari *server* yang ditampilkan pada *Listbox1* dapat dimuat dalam bentuk tabel pada *StringGrid1*. Kemudian dapat disimpan dengan format CSV. Tampilan bentuk tabel pada *StringGrid*1dan format CSV :



Identity	Latitude	Longitude
#Svr.Reply:->*B	21/05/2016	14:41:41.026
#Svr.Reply:->*B	21/05/2016	14:41:41.026
#Svr.Reply:->*B	21/05/2016	14:41:41.042
#Svr.Reply:->*B	21/05/2016	14:41:41.151
#Svr.Reply:->*B	21/05/2016	14:41:41.151
#Svr.Reply:->*B	21/05/2016	14:41:41.276
#Svr.Reply:->*B	21/05/2016	14:41:41.276
#Svr.Reply:->*B	21/05/2016	14:41:41.291
#Svr.Reply:->*B	21/05/2016	14:41:41.291
#Svr.Reply:->*B	21/05/2016	14:41:41.307

Gambar 4.4 Balasan dari *server* dalam bentuk tabel

	1	2	3	4	5
1	Identity	Latitude	Longitude	Date	TimeS
2	#Svr.Reply:->*Bima50wifi	-7.2874322	112.76797:	21/05/2016	23:26.4
3	#Svr.Reply:->*Bima50wifi	-7.2874322	112.76797:	21/05/2016	23:26.5
4	#Svr.Reply:->*Bima50wifi	-7.2874322	112.76797:	21/05/2016	23:26.5
5	#Svr.Reply:->*Bima50wifi	-7.2874322	112.76797:	21/05/2016	23:26.5
6	#Svr.Reply:->*Bima50wifi	-7.2874322	112.76797:	21/05/2016	23:26.6
7	#Svr.Reply:->*Bima50wifi	-7.2874322	112.76797:	21/05/2016	23:26.8

Gambar 4.5 *Save* dalam format CSV

4.1.1 Analisis Hasil Pengujian

Pengujian telah dilakukan, selanjutnya akan dianalisis hasil dari pengujian tersebut. Analisis tersebut meliputi apakah rancangan sistem *client-server* yang telah dibuat dapat berjalan sesuai perancangan atau tidak. Berikut ini Tabel 4-1 yang merupakan hasil pengujian dan analisis hasil pengujian :

Tabel 4-1 Hasil pengujian sistem *client-server*

No.	Pengujian	Hasil yang di dapat	Analisis
1	Penerimaan data oleh *server*	*Server* berhasil menerima data dari *client.*	Sistem telah sesuai dengan perancangan
2	Penyimpanan data dalam *database server*	Data dapat disimpan dalam bentuk *database* PHPMyAdmin	Sistem telah sesuai dengan perancangan
3	Penyimpanan balasan pengiriman data ke *client*	Balasan laporan dapat disimpan dalam format CSV	Sistem telah sesuai dengan perancangan

4.2 Pengujian Jaringan Komunikasi Terintegrasi AMC

Pengujian ini dilakukan untuk mengetahui kualitas jaringan yang sudah dibuat. Kualitas jaringan diukur mengikuti standar *Quality of Service* (QoS) adalah *packet loss* dan *throughput*.

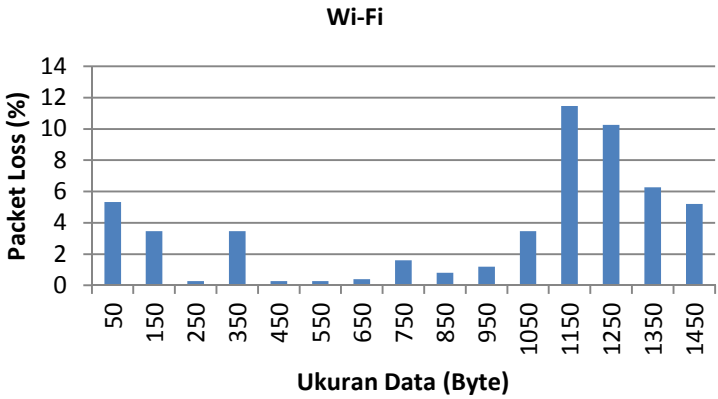
Pengujian dilakukan dengan mengirim paket data ke *server* sebesar 50 paket. Untuk jumlah *byte* nya sebesar 50-1500 *byte*. Pengujian ini menggunakan 2 buah laptop sebagai 2 *client*. Satu *client* melalui jaringan *backbone* dan satu *client* melalui jaringan seluler.

Jaringan yang diuji yaitu jaringan *backbone* dan jaringan seluler. Untuk mengukur kualitas jaringan pada jaringan *backbone,* digunakan jaringan Wi-Fi di tempat umum. Sedangkan jaringan seluler menggunakan salah satu jaringan operator seluler. Pengujian ini untuk mendapatkan ukuran data terbaik yang dapat diterima oleh *server* yang nantinya akan diterapkan pada sistem komunikasi terintegrasi angkutan massal cepat Surabaya.

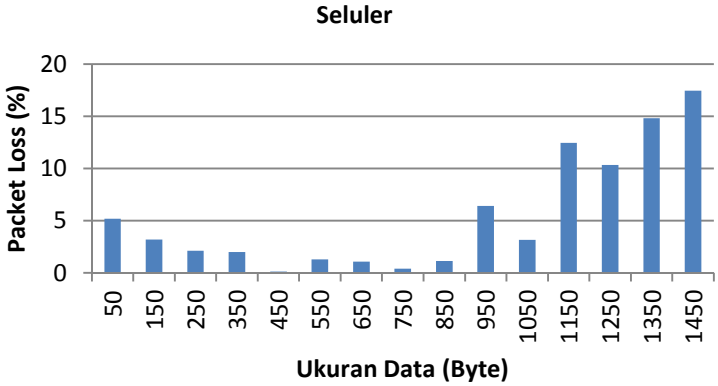
4.2.1 Pengujian dan Analisis *Packet Loss*

Pengujian *packet loss* digunakan untuk mengetahui paket data yang hilang sewaktu dilakukan pengiriman data dari *client* ke *server*. Pengujian ini dilakukan dengan mengirimkan data 50 paket secara terus-menerus selama 5 menit dengan jeda 15 detik per paket. Ukuran data yang dikirim ke *server* sebesar 50-1500 *byte*. Pengujian ini dilakukan dengan menggunakan dua laptop yang akan mengirim data secara bersamaan ke *server*. Satu laptop terkoneksi dengan jaringan Wi-Fi dan satu laptop terkoneksi dengan jaringan seluler. Hasil dari pengujian ini terdapat di tabel yang dilampirkan.

Pengujian ini menghasilkan perbandingan antara ukuran data (*byte*) yang dikirim dengan *packet loss* ketika pengiriman data menggunakan jaringan Wi-Fi dan jaringan seluler. Berikut hasil dari uji coba :



Gambar 4.6 Grafik *packet loss* pada Wi-Fi



Gambar 4.7 Grafik *packet loss* pada seluler

Pengujian ini menghasilkan nilai *packet loss* terbesar dan terkecil pada pengiriman data melalui jaringan Wi-Fi dan jaringan seluler. Pada jaringan Wi-Fi, *packet loss* terbesar dengan nilai sebesar 11.47% terletak pada ukuran data 1150 *byte*. *Packet loss* terkecil dengan nilai sebesar 0.267% pada ukuran data 250, 450, dan 550 *byte*. Pada jaringan seluler, *packet loss* terbesar dengan nilai sebesar 17.45% terletak pada ukuran data 1450 *byte*. Sedangkan untuk nilai *packet loss*

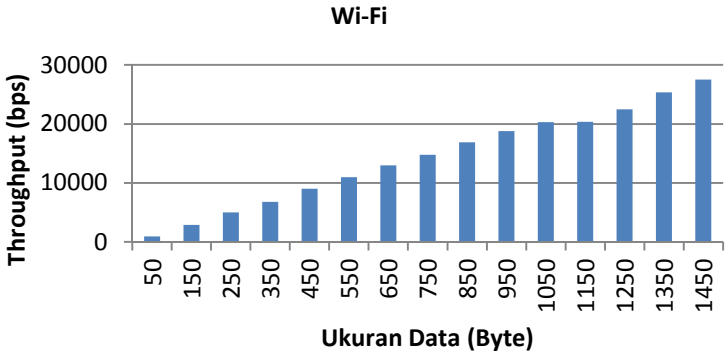
terkecil ketika pengiriman data melalui jaringan seluler sebesar 0.13% terletak pada ukuran data 450 *byte.*

Berdasarkan grafik hasil pengujian, dapat dianalisis bahwa timbulnya *packet loss* disebabkan oleh kepadatan trafik yang menimbulkan antrian sehingga beberapa paket data tidak diterima oleh server. *Packet loss* seluler lebih besar daripada *packet loss* Wi-Fi karena kapasitas *bandwidth* seluler lebih kecil daripada kapasitas *bandwidth* Wi-Fi.

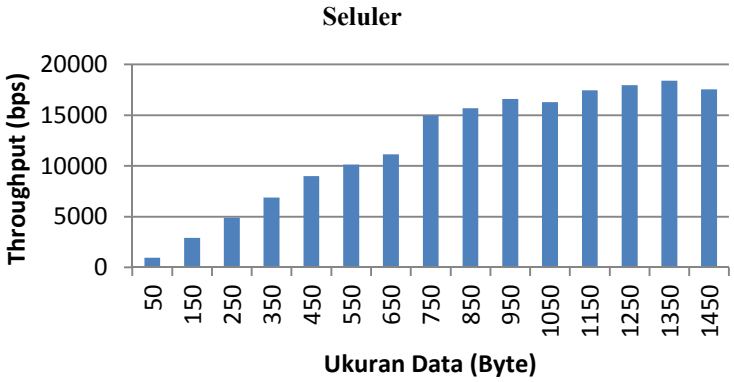
4.2.2 Pengujian dan Analisis *Throughput*

Perhitungan parameter ini akan digunakan untuk mendapatkan berapa *byte* atau jumlah panjang paket data yang optimal untuk dikirimkan pada *server*. Pengujian ini dilakukan dengan cara yang sama seperti pengujian *packet loss*, yaitu dengan mengirimkan data 50 paket secara terus menerus selama 5 menit dengan jeda 15 detik per 50 paket. Ukuran data yang dikirim ke *server* sebesar 50-1500 *byte*. Pengujian ini dilakukan dengan menggunakan dua laptop yang akan mengirim data secara bersamaan ke *server*. Satu laptop terkoneksi dengan jaringan Wi-Fi dan satu laptop terkoneksi dengan jaringan seluler.

Pengujian ini menghasilkan perbandingan antara ukuran data (*byte*) yang dikirim dengan *throughput* ketika pengiriman data menggunakan jaringan Wi-Fi dan jaringan seluler. Berikut hasil dari uji coba :



Gambar 4.8 *Throughput* pada Wi-Fi



Gambar 4.9*Throughput* pada seluler

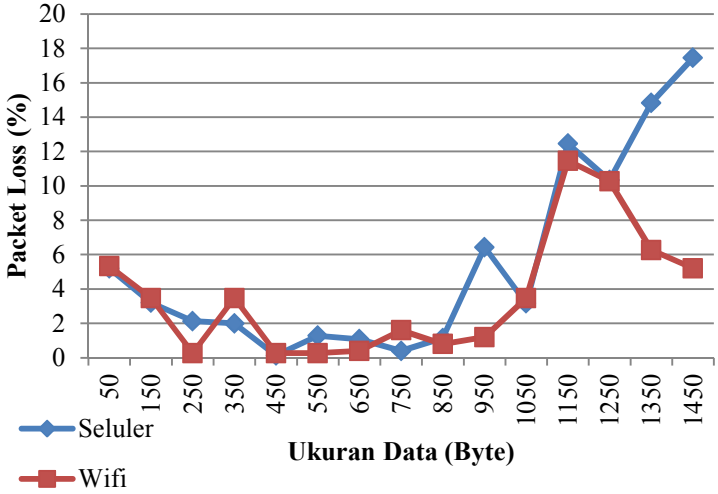
Pengujian ini menghasilkan nilai *throughput* ketika pengiriman data melalui jaringan Wi-Fi dan jaringan seluler. Pada jaringan Wi-Fi didapatkan hasil *throughput* paling besar terletak pada ukuran data yang paling besar dan yang terkecil terletak pada ukuran data yang paling kecil. Ukuran data terbesar yaitu 1450 *byte* dengan nilai *throughput* sebesar 27507.44 bps dan ukuran data terkecil yaitu 50 *byte* dengan nilai *throughput* sebesar 962.0533bps. Pada jaringan seluler didapatkan hasil *throughput* paling besar terletak pada ukuran data sebesar 1350 *byte* dengan nilai *throughput* sebesar 18406.77bps. Sedangkan *throughput* terkecil ketika pengiriman data melalui jaringan seluler terletak pada ukuran data paling kecil. Ukuran data paling kecil yaitu 50 *byte* dengan nilai *throughput* sebesar 963.52bps.

Dengan hasil pengujian tersebut, dapat dianalisis jika ukuran data semakin besar, maka nilai *throughput* yang dihasilkan semakin besar. Nilai *throughput* didapatkan dari pembagian antara jumlah ukuran data yang diterima dengan waktu pengiriman ke *server*. Namun pada jaringan seluler nilai *throughput* terbesar tidak terletak pada ukuran data tertinggi, hal ini terjadi karena pada 1450 *byte* banyak paket yang hilang.

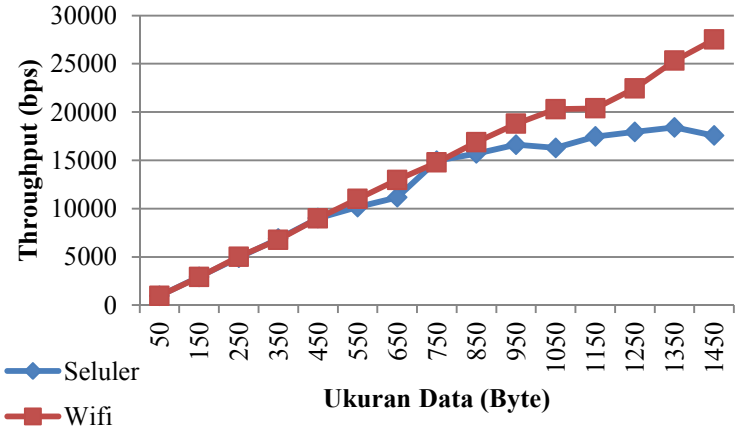
4.2.3 Rekomendasi Ukuran Data

Berdasarkan hasil pengujian *packet loss* dan *throughput*, didapatkan rekomendasi ukuran data yang bisa digunakan untuk sistem komunikasi terintegrasi angkutan massal cepat Surabaya. Ukuran data yang sebaiknya digunakan adalah sebesar 450 *byte*. Pemilihan ukuran

data sebesar 450 *byte* sebagai rekomendasi untuk sistem komunikasi terintegrasi angkutan massal cepat Surabaya dikarenakan ukuran data 450 *byte* memiliki nilai *packet loss* yang kecil, nilai *throughput* yang cukup bagus dan panjang data yang mencukupi (tidak terlalu pendek dan tidak terlalu panjang). Ukuran data 450 *byte* memiliki kualitas *packet loss perfect* berdasarkan standar versi TIPHON dan baik berdasarkan standar ITU-T G.114, baik melalui jaringan wi-fi maupun seluler. Nilai *packet loss* ukuran data 450 *byte* pada jaringan wifi sebesar 0.266667%, sedangkan pada jaringan seluler nilai *packet loss* nya sebesar 0.133333%. Nilai *throughput* yang dimiliki oleh ukuran data 450 *byte* sebesar 8992.347bps jika dikirim melalui jaringan wi-fi, sedangkan jika dikirim melalui jaringan seluler, nilai *throughput* nya sebesar 9004.373 bps. Ukuran data 450 *byte* yang tidak terlalu pendek dan tidak terlalu panjang sesuai dengan kebutuhan. Ukuran data 50, 150, 250, dan 350 memiliki panjang data kurang panjang. Agar lebih jelas, berikut grafik dan tabel hasil analisis :



Gambar 4.10 Perbandingan *Packet Loss* Wi-Fi dengan seluler

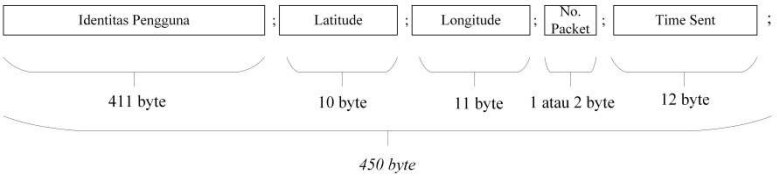


Gambar 4.11 Perbandingan *Throughput* Wi-Fi dengan seluler

Tabel 4-2 Kriteria ukuran data

Wifi	Ukuran Data (Byte)										
	450	550	650	750	850	950	1050	1150	1250	1350	1450
PL	√	√	√	√	√	√	√	x	x	x	x
Thr	√	√	√	√	√	√	√	√	√	√	√
Panjang	√	x	x	x	x	x	x	x	x	x	x
Seluler	Ukuran Data (Byte)										
	450	550	650	750	850	950	1050	1150	1250	1350	1450
PL	√	√	√	√	√	x	√	x	x	x	x
Thr	√	√	√	√	√	√	√	√	√	√	√
Panjang	√	x	x	x	x	x	x	x	x	x	x

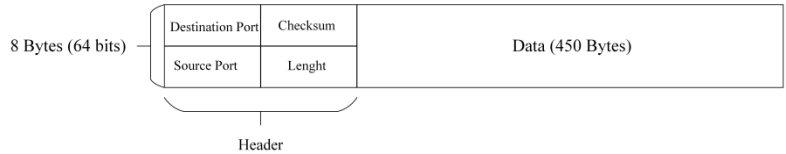
Berikut isi data yang dapat diterapkan sesuai dengan rekomendasi :



Gambar 4.12 Format data pengguna

Berdasarkan Gambar 4.12 ukuran data 450 *byte* dapat berisi identitas pengguna sebesar 411 *byte.* Dalam prakteknya nanti identitas pengguna dapat dipecah lagi menjadi beberapa bagian, seperti nomor ID, nama pengguna, dan klasifikasi pentarifan yang terdiri dari kelompok usia, kepala tiket, dan biaya yang harus dibayar. *Latitude* sebesar 10 *byte*. *Longitude* sebesar 11 *byte*. Nomor paket sebesar 1 atau 2 *byte*, *Time Sent* sebesar 12 *byte*. *Time sent* terdiri dari jam, menit, detik, dan mili detik (hh:mm:ss-zz). Pemisah tiap bagian data berupa titik koma (;) memiliki ukuran data sebesar 1 *byte*.

Dalam proses pengiriman data menggunakan UDP, data yang dikirimkan akan mengalami proses *encapsulation*. Proses enkapsulasi adalah sebuah proses yang mebuat satu jenis paket data jaringan menjadi jenis data lainnya. Enkapsulasi terjadi ketika sebuah protokol yang berada pada lapisan yang lebih rendah menerima data dari protokol yang berada pada lapisan yang lebih tinggi dan meletakkan data ke format data yang dipahami oleh protokol tersebut. Enkapsulasi terhadap pesan-pesan UDP oleh protokol IP dilakukan dengan menambahkan *header* IP dengan protokol IP nomor 17 (0x11). Pesan UDP dapat memiliki besar maksimum 65507 *byte* : 65535 (216)-20 (ukuran terkecil dari *header* IP)-8 (ukuran dari *header* UDP) *byte. Overhead* lebih kecil karena *header* yang digunakan untuk enkapsulasi lebih kecil. UDP memiliki *overhead* sebesar 8 *bytes.* Berikut ini gambar mengenai penambahan *overhead* :



Gambar 4.13 Penambahan *header*

Rekomendasi ukuran data ini dapat berlaku pada kondisi seperti pada pengujian ini.

LAMPIRAN

A. Lembar Pengesahan Proposal

JurusanTeknik Elektro
Fakultas Teknologi Industri – ITS

TE141599 TUGAS AKHIR – 4 SKS

Nama Mahasiswa	: Tiffany Maliati Khumairoh Afandi
Nomor Pokok	: 2212100182
Bidang Studi	: Teknik Telekomunikasi Multimedia
Tugas Diberikan	: Semester Genap Th. 2015/2016
Dosen Pembimbing	: 1. Dr. Ir. AchmadAffandi, DEA
	2. Ir. Djoko Suprajitno Rahardjo, MT
Judul Tugas Akhir	: ***Testbed* Komunikasi Terintegrasi untuk Angkutan Massal Cepat Surabaya**
	(*Integrated Communications Testbed for Mass Rapid Transport Surabaya*)

10 FEB 2016

UraianTugasAkhir :
Kota Surabaya memiliki impian menjadi kota pintar yang dapat mendukung dan mengoptimalkan semua moda transportasi, sehingga dapat memberikan layanan yang maksimal kepada masyarakatnya dengan berbasis Teknologi Informasi Komunikasi (TIK). Hal tersebut dapat terwujud dalam *Intelligent Transportation System* (Sistem Transportasi Cerdas). Salah satu bagian dari *Intelligent Transportation System*, yaitu *Advanced Public Transportation Systems (APTS)*. APTS adalah sistem transportasi publik canggih yang memungkinkan armada seperti kereta, *trem*, dan *bus* dapat melaporkan posisi mereka AVL (*automatic vehicle location*), sehingga penumpang mendapat informasi status *real-time* armada dan bagi pengelola dapat memonitor keberadaan aset mereka. Impian tersebut dapat diwujudkan dengan adanya manajemen armada yang baik. Pada manajemen armada terdapat *On-Board Unit*(OBU) yang akan dipasang pada masing-masing armada. Data dari *On-Board Unit* tersebut akan diteruskan melalui jaringan seluler, kemudian di kirim ke *Control Center Room* (CC-ROOM). Untuk menguji sistem tersebut diperlukan suatu *testbed* yang akan di uji coba pada sejumlah titik OBU.Selain uji coba pada OBU,juga akan dilakukan uji coba pada sistem *Passenger Information Board* (PIB) yang terdapat di halte. Jaringan komunikasi yang digunakan pada sistem PIB dalah jaringan *backbone* atau *wireless fidelity* (Wi-Fi) yang tersedia pada halte. Tugas akhir ini akan menghasilkan *testbed* dari *prototype* sistem

Dosen Pembimbing I,

Dr. Ir. AchmadAffandi, DEA.
NIP. 196510141990021001

Dosen Pembimbing II,

Ir. Djoko Suprajitno Rahardjo, MT
NIP.195506221987011001

Mengetahui,
Jurusan Teknik Elektro FTI-ITS
Ketua,

Dr. Eng.Ardyono Priyadi, ST., M.Eng
NIP. 19730927 199803 1 004

Menyetujui,
Bidang Studi Telekomunikasi Multimedia
Kordinator,

Dr. Ir. Endroyono, DEA
NIP. 1965040419910210

1

B. Tabel Hasil Pengujian

1. Pengujian *Packet Loss*

- Jaringan Wi-Fi

Byte	Packet Loss	%
50	40	5.333333
150	26	3.466667
250	2	0.266667
350	26	3.466667
450	2	0.266667
550	2	0.266667
650	3	0.4
750	12	1.6
850	6	0.8
950	9	1.2
1050	26	3.466667
1150	86	11.46667
1250	77	10.26667
1350	47	6.266667
1450	39	5.2

- Jaringan Seluler

Byte	Packet Loss	%
50	39	5.2
150	24	3.2
250	16	2.133333
350	15	2
450	1	0.133333
550	9	1.285714

Byte	*Packet Loss*	%
650	7	1.076923
750	3	0.4
850	8	1.142857
950	45	6.428571
1050	19	3.166667
1150	81	12.46154
1250	62	10.33333
1350	89	14.83333
1450	96	17.45455

2. Pengujian *Throughput*
   - Jaringan Wi-Fi

Byte	Datasize	Durasi	Throughput(byte/ms)	bps
50	36077	300000	0.120256667	962.0533
150	109190	300000	0.363966667	2911.733
250	187614	300000	0.62538	5003.04
350	253989	300000	0.84663	6773.04
450	337213	300000	1.124043333	8992.347
550	412013	300000	1.373376667	10987.01
650	486162	300000	1.62054	12964.32
750	554103	300000	1.84701	14776.08
850	633009	300000	2.11003	16880.24
950	704557	300000	2.348523333	18788.19
1050	760791	300000	2.53597	20287.76
1150	764142	300000	2.54714	20377.12
1250	841801	300000	2.806003333	22448.03
1350	949618	300000	3.165393333	25323.15
1450	1031529	300000	3.43843	27507.44

- Jaringan Seluler

Byte	Datasize	Durasi	Throughput (byte/ms)	bps
50	36132	300000	0.12044	963.52
150	109491	300000	0.36497	2919.76
250	184099	300000	0.613663333	4909.307
350	257850	300000	0.8595	6876
450	337664	300000	1.125546667	9004.373
550	380615	300000	1.268716667	10149.73
650	418476	300000	1.39492	11159.36
750	560862	300000	1.86954	14956.32
850	588766	300000	1.962553333	15700.43
950	622787	300000	2.075956667	16607.65
1050	610530	300000	2.0351	16280.8
1150	654822	300000	2.18274	17461.92
1250	672947	300000	2.243156667	17945.25
1350	690254	300000	2.300846667	18406.77
1450	658660	300000	2.195533333	17564.27

C. *Listing* Program Borland Delphi 7

```
unit UTestbedAMC;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
    Forms,
  Dialogs, IdBaseComponent, IdComponent, IdUDPBase, IdUDPServer,
    StdCtrls,
  IdSocketHandle, IdUDPClient, ExtCtrls, Grids, Menus;

type
  TForm1 = class(TForm)
```

```
Edit1: TEdit;
Button1: TButton;
Label1: TLabel;
Button2: TButton;
Label2: TLabel;
Button3: TButton;
Label3: TLabel;
Edit2: TEdit;
Label4: TLabel;
Edit3: TEdit;
IdUDPServer1: TIdUDPServer;
IdUDPServer2: TIdUDPServer;
Timer1: TTimer;
Label5: TLabel;
Label6: TLabel;
MainMenu1: TMainMenu;
FileSave: TMenuItem;
FileExit: TMenuItem;
ListBox1: TListBox;
SG1: TStringGrid;
ListBox2: TListBox;
Label7: TLabel;
Edit4: TEdit;
FileDB: TMenuItem;
SaveDialog1: TSaveDialog;
Label8: TLabel;
Edit5: TEdit;
SG2: TStringGrid;
FileDB2: TMenuItem;
FileSave1: TMenuItem;
SaveDialog2: TSaveDialog;
FileAbout: TMenuItem;
Memo1: TMemo;
procedure FormCreate(Sender: TObject);
procedure IdUDPServer1UDPRead(Sender: TObject; AData:
  TStream;
  ABinding: TIdSocketHandle);
procedure Button1Click(Sender: TObject);
```

```
    procedure IdUDPServer2UDPRead(Sender: TObject; AData:
      TStream;
      ABinding: TIdSocketHandle);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure FileExitClick(Sender: TObject);
    procedure FileSaveClick(Sender: TObject);
    procedure FileDBClick(Sender: TObject);
    procedure FileDB2Click(Sender: TObject);
    procedure FileSave1Click(Sender: TObject);
     procedure Tes;
    procedure FileAboutClick(Sender: TObject);

  private
  procedure ParseRecord(sRecord: string; Row: integer);
  function PosFirstSep(sRecord: string): integer;
  procedure ParseRecord1(sRecord1: string; Row1: integer);
  function PosFirstSep1(sRecord1: string): integer;
  procedure AddRecToList1(Row: integer);
  procedure AddRecToList(Row: integer);

  //    procedure Tes(i:integer);
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  abc:string;
  JumlahData:integer;
  balikin:string;



implementation

{$R *.dfm}
```

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  IdUDPServer1.Active:=True;
  IdUDPServer2.Active:=True;
end;
////////////////////////////////////////////////////////////////////////
procedure TForm1.IdUDPServer1UDPRead(Sender: TObject; AData:
    TStream;
  ABinding: TIdSocketHandle);
var
  s:ShortString;
  dataSize:Byte;
  waktu:string;
  a:tdatetime;
begin
  a:=now;
  DateTimeToString(waktu, 'hh:nn:ss.zzz', a);
  balikin:=Abinding.PeerIP;
  dataSize:=AData.Size;
  Memo1.Lines.Add('ukuran data = '+IntToStr(Adata.Size));
  AData.Read(s[1],dataSize);
  s[0]:=char(dataSize);

    ListBox1.Items[0]:='Identity;Latitude;Longitude;Number;TimeS;IP
    Client;Date;TimeR';
  ListBox1.Items.Add(s
    +ABinding.PeerIP+';'+DateToStr(Now)+';'+waktu);

end;
 ////////////////////////////////////////////////////////////////////////


procedure TForm1.Button1Click(Sender: TObject);
var
  i,A,c:integer;
  val,b,d:string;
  waktu:string;
  h:tdatetime;
begin
```

```
 h:=now;
 A:=0;
 DateTimeToString(waktu, 'hh:nn:ss.zzz', h);
 abc:=Edit2.Text;
 JumlahData:=StrtoInt(Edit3.Text);
 Memo1.Lines.Add('Start='+FormatDateTime('hh:nn:ss.zzz',Now));
 for i:=1 to JumlahData do
 begin
  h:=now;
  DateTimeToString(waktu, 'hh:nn:ss.zzz', h);
  b :=Edit5.text;
  c:= pos(',',b);
  d:=copy(b,1,c-1);
  delete(b,1,c);
  val:=Edit1.Text+';'+d+';'+b+ ';'+IntToStr(i)+ ';'+waktu+ ';';
  IdUDPServer1.Send(abc,5000, val);
  sleep(100);
  A:=A+Length(val)*SizeOf(Char);
  Application.ProcessMessages;
 end;
 Memo1.Lines.Add('Stop='+FormatDateTime('hh:nn:ss.zzz',Now));
Edit4.Text:=IntToStr(A)+' byte';
Tes;
end;
////////////////////////////////////////////////////////////////////
procedure TForm1.IdUDPServer2UDPRead(Sender: TObject; AData:
    TStream;
  ABinding: TIdSocketHandle);
var
  s:ShortString;
  dataSize:Byte;
begin
  dataSize:=AData.Size;
  AData.Read(s[1],dataSize);
  s[0]:=char(dataSize);
  ListBox2.Items[0]:='Information';
  ListBox2.Items.Add(s);
end;
////////////////////////////////////////////////////////////////////////
```

```
//procedure TForm1.Tes(i:integer);
procedure TForm1.Tes;
var
  waktu,sRecord:string;
  h,i:integer;
  a:tdatetime;
  Row:integer;
begin
  a:=now;
  DateTimeToString(waktu, 'hh:nn:ss.zzz', a);

 for i:=1 to JumlahData do
  //begin
  IdUDPserver2.Send(balikin,5001,'Data ke '+InttoStr(i)+' telah
    dikirim');//datetimetostr
  sleep(100);
  Application.ProcessMessages;

end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
Label5.Caption:=FormatDateTime('hh:nn:ss-zzz',Now);
end;
////////////////////////////////////////////////////////////////////////////
procedure TForm1.Button2Click(Sender: TObject);
begin
ListBox1.Clear;
end;
////////////////////////////////////////////////////////////////////
procedure TForm1.Button3Click(Sender: TObject);
begin
ListBox2.Clear;
end;
////////////////////////////////////////////////////////////////

procedure TForm1.FileExitClick(Sender: TObject);
begin
  Close;  // close main window of application
```

```
end;
////////////////////////////////////////////////////////////////////////
procedure TForm1.FileSaveClick(Sender: TObject);
var
  Row: integer;
begin
saveDialog1.DefaultExt:='csv';
if saveDialog1.Execute then begin
  ListBox1.Clear;                    // 1. clear listbox
  for Row := 0 to SG1.RowCount - 1 do   // 2. for every record...
                                 // (row counting starts at 0!)
    AddRecToList(Row);                 // add to listbox
  // Use same filename as when loading.
  // For now, let's simply hard-code it.
  //FileName1 := 'E:\delphy\delphy - Copy\testbed\Testbed2.csv';
  ListBox1.Items.SaveToFile(saveDialog1.FileName); // 3. save as csv-
     file
  end
 else
 ShowMessage('Save file was cancelled');
 saveDialog1.Free;
end;
////////////////////////////////////////////////////////////////////////
procedure TForm1.ParseRecord(sRecord: string; Row: integer);
var
  Col, PSep1: integer;
  sField: string;
begin
  Col := 0;                      // (1) first column of stringgrid
  repeat
    sRecord := Trim(sRecord);       // (2) leading & trailing blanks
    PSep1 := PosFirstSep(sRecord);  // (3) first separator
    // (4) Extract first field
    if PSep1 > 0 then              // separator found
      sField := Copy(sRecord, 1, PSep1 - 1)
    else                          // no separator: this is last field
      sField := sRecord;
    // (5) Clean up field value
    sField := Trim(sField);        // leading & trailing blanks
```

```
    if sField[1] = '"' then begin  // " at beginning?
      Delete(sField, 1, 1);                // delete "
      if sField[Length(sField)] = '"' then  // " at end?
        Delete(sField, Length(sField), 1);  // delete "
      sField := Trim(sField);   // leading & trailing blanks
    end;
    // Replace each "" pair with one " character
    sField := StringReplace(sField, '""', '"', [rfReplaceAll]);
    SG1.Cells[Col, Row] := sField; // (6) put field in stringgrid
    if PSep1 > 0 then begin        // (7) more fields?
      Delete(sRecord, 1, PSep1);   // delete field from record string
      Col := Col + 1;              // next column
    end;
  until PSep1 = 0;           // (8) if more fields, then repeat
end;
////////////////////////////////////////////////////////////////////////
function TForm1.PosFirstSep(sRecord: string): integer;
var
  i, PosSecondDoubleQ: integer;
begin
  if sRecord[1] <> '"' then     // (1) no " at beginning?
    Result := Pos(';', sRecord)   // done
  else begin                    // (2) else if starts with "
    // prepare loop
    Result := 0;
    PosSecondDoubleQ := 0;
    i := 2;

    // (3) Loop until end of string, or until separator found
    while (i <= Length(sRecord)) and (Result = 0) do begin
      if PosSecondDoubleQ > 0 then begin // (3.1)
        if sRecord[i] = ';' then // (3.1.1) comma found?
          Result := i              // done: this is the separator comma
        else    // (3.1.2) Skip pairs of double quotes
          if (sRecord[i] = '"') and (PosSecondDoubleQ = i - 1) then
            PosSecondDoubleQ := 0; // reset pos. of second double quote
      end
      else // (3.2) no second double quote found yet
        if sRecord[i] = '"' then  // double quote found?
```

```
        PosSecondDoubleQ := i; // remember second double
      inc(i); // (3.3) prepare for next character
    end; // (4) end of the loop, go back to (3)
  end;
end;
////////////////////////////////////////////////////////////////////
procedure TForm1.AddRecToList(Row: integer);
var
  Col: integer;
  sField, sRecord: string;
begin
  sRecord := ''; // (1) start with empty record
  Col := 0;      // (2) start at column number 0
  repeat
    sField := SG1.Cells[Col, Row];   // (3) get contents of cell
    // (4) Replace " with ""
    sField := StringReplace(sField, '"', '""', [rfReplaceAll]);
    // (5) If field contains double quote, space or comma, then...
    // ...surround with double quotes
    if (Pos('"', sField) > 0) or
       (Pos(' ', sField) > 0) or
       (Pos(';', sField) > 0) then
      sField := '"' + sField + '"';
    sRecord := sRecord + sField;    // (6) add field to record
    if Col < SG1.ColCount - 1 then  // (7) more fields?
      sRecord := sRecord + ';';       // add comma to record
    Col := Col + 1;                 // (8) increment column counter
  until Col = SG1.ColCount;       // (9) if more columns, repeat
  ListBox1.Items[Row] := sRecord; // (10) put record-string in listbox
end;

////////////////////////////////////////////////////////////////////////////
procedure TForm1.FileDBClick(Sender: TObject);
var
  sRecord: string;
  Row: integer;
begin
  SG1.RowCount := ListBox1.Items.Count;
```

```
  // for every record... ( count starts at 0 ! )
  for Row := 0 to ListBox1.Items.Count - 1 do begin
    sRecord := ListBox1.Items[Row];
    ParseRecord(sRecord, Row);            // 4.
  end;
  // 5. Select first "data" cell
  SG1.Row := 1;
  SG1.Col := 0;
  SG1.SetFocus;

end;
////////////////////////////////////////////////////////////////////////
procedure TForm1.ParseRecord1(sRecord1: string; Row1: integer);
var
  Col, PSep1: integer;
  sField: string;
begin
  Col := 0;                       // (1) first column of stringgrid
  repeat
    sRecord1 := Trim(sRecord1);       // (2) leading & trailing blanks
    PSep1 := PosFirstSep1(sRecord1);  // (3) first separator
    // (4) Extract first field
    if PSep1 > 0 then               // separator found
      sField := Copy(sRecord1, 1, PSep1 - 1)
    else                          // no separator: this is last field
      sField := sRecord1;
    // (5) Clean up field value
    sField := Trim(sField);        // leading & trailing blanks
    if sField[1] = '"' then begin  // " at beginning?
      Delete(sField, 1, 1);               // delete "
      if sField[Length(sField)] = '"' then  // " at end?
        Delete(sField, Length(sField), 1);  // delete "
      sField := Trim(sField);   // leading & trailing blanks
    end;
    // Replace each "" pair with one " character
    sField := StringReplace(sField, '""', '"', [rfReplaceAll]);
    SG2.Cells[Col, Row1] := sField; // (6) put field in stringgrid
    if PSep1 > 0 then begin        // (7) more fields?
```

```
    Delete(sRecord1, 1, PSep1);   // delete field from record string
    Col := Col + 1;               // next column
   end;
  until PSep1 = 0;           // (8) if more fields, then repeat
end;
////////////////////////////////////////////////////////////////////
function TForm1.PosFirstSep1(sRecord1: string): integer;
var
  i, PosSecondDoubleQ: integer;
begin
  if sRecord1[1] <> '"' then     // (1) no " at beginning?
    Result := Pos(';', sRecord1)   // done
  else begin                   // (2) else if starts with "
    // prepare loop
    Result := 0;
    PosSecondDoubleQ := 0;
    i := 2;

    // (3) Loop until end of string, or until separator found
    while (i <= Length(sRecord1)) and (Result = 0) do begin
      if PosSecondDoubleQ > 0 then begin // (3.1)
        if sRecord1[i] = ';' then // (3.1.1) comma found?
          Result := i             // done: this is the separator comma
        else    // (3.1.2) Skip pairs of double quotes
          if (sRecord1[i] = '"') and (PosSecondDoubleQ = i - 1) then
            PosSecondDoubleQ := 0; // reset pos. of second double quote
      end
      else // (3.2) no second double quote found yet
        if sRecord1[i] = '"' then  // double quote found?
          PosSecondDoubleQ := i; // remember second double
      inc(i); // (3.3) prepare for next character
    end; // (4) end of the loop, go back to (3)
  end;
end;
////////////////////////////////////////////////////////////

procedure TForm1.FileDB2Click(Sender: TObject);
var
  sRecord1: string;
```

```
  Row1: integer;
begin
  SG2.RowCount := ListBox2.Items.Count;    // 3.

  for Row1 := 0 to ListBox2.Items.Count -1 do begin
    sRecord1 :=ListBox2.Items[Row1];
    ParseRecord1(sRecord1, Row1);
    end;
  SG2.Row := 1;
  SG2.Col := 0;
  SG2.SetFocus;
end;
////////////////////////////////////////////////////////
procedure TForm1.FileSave1Click(Sender: TObject);
var
  Row: integer;
begin
saveDialog2.DefaultExt:='csv';
if saveDialog2.Execute then begin
  ListBox2.Clear;                    // 1. clear listbox
  for Row := 0 to SG2.RowCount - 1 do   // 2. for every record...
                                 // (row counting starts at 0!)
    AddRecToList1(Row);                  // add to listbox
  // Use same filename as when loading.
  // For now, let's simply hard-code it.
  //FileName1 := 'E:\delphy\delphy - Copy\testbed\Testbed2.csv';
  ListBox2.Items.SaveToFile(saveDialog2.FileName); // 3. save as csv-
     file
  end
 else
 ShowMessage('Save file was cancelled');
 saveDialog2.Free;
end;
////////////////////////////////////////////////
procedure TForm1.AddRecToList1(Row: integer);
var
  Col: integer;
  sField, sRecord: string;
begin
```

```
  sRecord := ''; // (1) start with empty record
  Col := 0;      // (2) start at column number 0
  repeat
    sField := SG2.Cells[Col, Row];   // (3) get contents of cell
    // (4) Replace " with ""
    sField := StringReplace(sField, '"', '""', [rfReplaceAll]);
    // (5) If field contains double quote, space or comma, then...
    // ...surround with double quotes
    if (Pos('"', sField) > 0) or
       (Pos(' ', sField) > 0) or
       (Pos(';', sField) > 0) then
      sField := '"' + sField + '"';
    sRecord := sRecord + sField;    // (6) add field to record
    if Col < SG2.ColCount - 1 then  // (7) more fields?
      sRecord := sRecord + ';';       // add comma to record
    Col := Col + 1;                 // (8) increment column counter
  until Col = SG1.ColCount;       // (9) if more columns, repeat
  ListBox2.Items[Row] := sRecord; // (10) put record-string in listbox
end;

procedure TForm1.FileAboutClick(Sender: TObject);
begin
showmessage('Tiffany 2212100182 tugas akhir 2016');
end;
end.
```

BAB V
PENUTUP

5.1 Kesimpulan

Setelah dilakukan perancangan dan pengujian sistem dan jaringan komunikasi terintegrasi Angkutan Massal Cepat Surabaya, maka secara keseluruhan dapat diambil kesimpulan sebagai berikut:

1. Pengujian sistem *client-server* mengenai penerimaan data oleh *server,* penyimpanan data data dalam *database server*, dan penyimpanan data balasan ke *client* telah sesuai rancangan. Sistem yang disimulasikan mampu memenuhi kebutuhan selama proses pengiriman dan penerimaan.
2. Pengujian *packet loss* terkecil pengiriman data melalui jaringan wi-fi terletak pada ukuran data 250, 450, dan 550 *byte* sebesar 0.26667% (*perfect*). Sedangkan pada jaringan seluler nilai *packet loss* terkecil terletak pada ukuran data 450 *byte* sebesar 0.13333% (*perfect*).
3. Nilai *throughput* berbanding lurus dengan ukuran data yang diterima.
4. Rekomendasi ukuran data yang dapat diterapkan pada sistem komunikasi terintegrasi angkutan massal cepat Surabaya, yaitu sebesar 450 *byte*. Rekomendasi ini berlaku pada kondisi seperti pengukuran ini.

5.2 Saran

Adapun hal-hal yang masih bisa dikembangkan dari *testbed* komunikasi terintegrasi Angkutan Massal Cepat adalah :

1. Pengembangan aplikasi yang dibuat menggunakan Borland Delphi 7 untuk *testbed* komunikasi terintegrasi AMC
2. Kedepannya dapat dilakukan pengukuran terhadap *delay* dan *jitter.*
3. Pengukuran dilakukan pada jaringan yang lebih bervariasi dan banyak.

DAFTAR PUSTAKA

[1] Farhana, Nurul. "Pengembangan Aplikasi *Remote* Spesifikasi Dekstop Berbasis *Client Server*," Universitas Islam Negeri Syarif Hidayatullah, Jakarta, 2011.

[2] Yuliantoro, Prasetyo. "Rancang Bangun Protokol *E-Ticketing*," Institut Teknologi Sepuluh Nopember, Surabaya, 2015.

[3] Affandi, Achmad dan tim. "Laporan Akhir Kajian Sistem dan Teknologi IT dalam Rangka Integrasi Angkutan Massal Cepat *Trunk* dan *Feeder* AMC Surabaya," Dinas Perhubungan Kota Surabaya, Surabaya, 2015.

[4] Faesal, Andris. "Pemrograman *Client-server* dengan Borland Delphi 7 dan My SQL," STMIK Bumigora, Mataram, 2012.

[5] Indrajaya, Muhammad Aristo. "Perancangan *Geographic Information System* untuk *Tracking* dan *Routing* Menggunakan Algoritma Djikstra pada Kendaraan Angkutan Umum," Institut Teknologi Sepuluh Nopember, Surabaya, 2015.

[6] Xue, Ming. Zhu, Changjun. "*The Socket Programming and Software Design for Communication Based on Client/Server*," IEEE Pasific-Asia Conference on Circuits, Communication and System, Chengdu, 2009.

[7] Sastriyana, Nym Yuni. "Perancangan dan Analisis Interkoneksi Jaringan pada *Electronic Toll Collection* Berbasis *Radio Frequency Identification* (RFID)," Institut Teknologi Sepuluh Nopember, Surabaya, 2010.

[8] Tiphon, "*Telecommunications and Internet Protocol Harmonization Over Network (TIPHON) General aspects of Quality of Service (QoS)*," DTR/TIPHON-05006 (cb0010cs.pdf), 1999.

[9] ITU-T Series G, "*Transmission system and media, digital system and network*," 2003.

[10] Halim, Ansarullah. Affandi, Achmad. Maulidiyanto, Achmad. "Analisa Trafik Komunikasi Data pada Kanal Propagasi VHF Bergerak untuk Kelautan," Institut Teknologi Sepuluh Nopember, Surabaya, 2011.

RIWAYAT HIDUP

Tiffany Maliati Khumairoh Afandi, lahir di Surabaya pada tanggal 17 Februari 1994. Menamatkan pendidikan dasarnya di SD Nurul Ulum Surabaya pada tahun 2006 lalu meneruskan di SMP Negeri 1 Surabaya dan lulus pada tahun 2009. Kemudian melanjutkan pendidikan menengah atas di SMA Negeri 5 Surabaya dan lulus pada tahun 2012. Setelah itu melanjutkan pendidikan strata satu (S1) di jurusan Teknik Elektro Institut Teknologi Sepuluh Nopember Surabaya pada tahun 2012 hingga 2016 dengan program studi Telekomunikasi Multimedia. Penulis dapat dihubungi melalui alamat email ini : tiffanymaliati@gmail.com.