



# INSTITUT TEKNOLOGI BANDUNG

## PROGRAM STUDI TEKNIK ELEKTRO

JALAN GANESHA NO. 10 Gedung Labtek V Lantai 2 ☎ (022)2508135-36, 📠 (022)250 0940  
BANDUNG 40132

### Dokumentasi Produk Tugas Akhir

#### Lembar Sampul Dokumen

Judul Dokumen	<b>TUGAS AKHIR TEKNIK ELEKTRO:</b> <i>Fleet Monitoring and Control System pada Guided Bus</i>
Jenis Dokumen	<b>DESAIN</b> <small>Catatan: Dokumen ini dikendalikan penyebarannya oleh Prodi Teknik Elektro ITB</small>
Nomor Dokumen	<b>B300-02-TA1617.01.094</b>
Nomor Revisi	<b>02</b>
Nama File	<b>B300-02-TA1617.01.094</b>
Tanggal Penerbitan	<b>12 May 2017</b>
Unit Penerbit	<b>Prodi Teknik Elektro – ITB</b>
Jumlah Halaman	<b>33</b> (termasuk lembar sampul ini)

Data Pengusul				
Pengusul	Nama	Ali Zaenal Abidin	Jabatan	<b>Mahasiswa</b>
	Tanggal	4 Mei 2017	Tanda Tangan	
	Nama	Shah Dehan Lazuardi	Jabatan	<b>Mahasiswa</b>
	Tanggal	4 Mei 2017	Tanda Tangan	
	Nama	Aulia Hening Darmasti	Jabatan	<b>Mahasiswa</b>
	Tanggal	4 Mei 2017	Tanda Tangan	
Pembimbing	Nama	Arief Syaichu R.	Tanda Tangan	
	Tanggal	4 Mei 2017		
Lembaga Program Studi Teknik Elektro Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung				
Alamat Labtek V, Lantai 2, Jalan Ganesha no. 10, Bandung Telepon : +62 22 250 2260      Faks : +62 22 253 4222      Email: stei@stei.itb.ac.id				

## DAFTAR ISI

<b>DAFTAR ISI.....</b>	<b>2</b>
<b>CATATAN SEJARAH PERBAIKAN DOKUMEN .....</b>	<b>4</b>
<b>FLEET MONITORING AND CONTROL SYSTEM PADA GUIDED BUS.....</b>	<b>5</b>
<b>1    PENGANTAR .....</b>	<b>5</b>
1.1    RINGKASAN ISI DOKUMEN .....	5
1.2    TUJUAN PENULISAN DAN APLIKASI/KEGUNAAN DOKUMEN .....	5
1.3    REFERENSI .....	5
1.4    DAFTAR SINGKATAN.....	6
<b>2    DESKRIPSI UMUM.....</b>	<b>6</b>
2.1    DEFINISI, FUNGSI DAN SPESIFIKASI .....	6
2.2    FLOWCHART SISTEM.....	8
2.2.1 <i>Flowchart Algoritma pada GUI</i> .....	8
2.3    DIAGRAM BLOK INPUT-OUTPUT .....	8
2.3.1 <i>Hardware</i> .....	8
2.3.2 <i>Graphical User Interface di Control Station</i> .....	9
2.4    DATA FLOW DIAGRAM.....	11
2.4.1 <i>DFD Level 0</i> .....	11
2.4.2 <i>DFD level 1</i> .....	12
2.4.3 <i>DFD level 2</i> .....	12
<b>3    ARSITEKTUR SISTEM .....</b>	<b>14</b>
3.1    PILIHAN ARSITEKTUR PERTAMA .....	14
3.2    PILIHAN ARSITEKTUR KEDUA .....	15
3.3    PILIHAN ARSITEKTUR KETIGA .....	15
3.4    ARSITEKTUR SISTEM KESELURUHAN .....	16
<b>4    DESAIN.....</b>	<b>16</b>
4.1    ALGORITMA PENJADWALAN .....	16
4.2    DESAIN GRAPHICAL USER INTERFACE.....	19
4.2.1 <i>Peta geografis</i> .....	19
4.2.2 <i>Peta diagramatis</i> .....	21
4.2.3 <i>Pengiriman komando</i> .....	22
4.2.4 <i>Layout dan Platform GUI</i> .....	22
4.3    DESAIN SERVER DAN PROTOKOL KOMUNIKASI.....	23
4.3.1 <i>Protokol Komunikasi</i> .....	23
4.3.2 <i>Server</i> .....	24
4.4    DESAIN HARDWARE .....	26
4.4.1 <i>Modul Akuisisi Data ECU</i> .....	26
4.4.2 <i>Modul Komunikasi</i> .....	27
4.4.3 <i>Modul GPS</i> .....	28
4.4.4 <i>Kontroller</i> .....	29
4.4.5 <i>Display</i> .....	30
4.4.6 <i>Sumber Daya Listrik</i> .....	30
4.4.7 <i>Desain Packaging</i> .....	30
4.5    DESAIN SISTEM KESELURUHAN.....	31

4.6	DESAIN SIMULASI ALGORITMA PENJADWALAN .....	32
4.6.1	<i>Aplikasi Dummy Data</i> .....	32
4.6.2	<i>Server</i> .....	32
4.6.3	<i>Graphical User Interface (GUI)</i> .....	33

## Catatan Sejarah Perbaikan Dokumen

VERSI, TGL, OLEH	PERBAIKAN
2, 4 Mei 2017, Ali Zaenal A. Shah Dehan L. Aulia Hening D.	Menambahkan perangkat untuk komunikasi CAN dengan ECU pada <i>guided bus</i> Menambahkan desain <i>packaging</i> Penambahan pilihan <i>broker server</i>

# ***Fleet Monitoring and Control System pada Guided Bus***

## **1 Pengantar**

### **1.1 RINGKASAN ISI DOKUMEN**

Dokumen ini berisi tentang perancangan desain *Fleet Monitoring and Control System* secara detil, baik subsistem maupun secara keseluruhan. Dalam dokumen ini dijelaskan mengenai desain perancangan produk yang dibagi menjadi beberapa bagian, diantaranya penjelasan mengenai spesifikasi dan fungsi sistem, diagram blok sistem-subsistem, flowchart sistem-subsistem, data flow diagram, desain alternatif arsitektur sistem, hingga desain detail dari setiap sistem dan subsistem. Pada dokumen ini juga dilakukan *brainstorming* dalam menentukan desain alternatif yang dipilih untuk masing-masing subsistem.

### **1.2 Tujuan Penulisan dan Aplikasi/Kegunaan Dokumen**

Tujuan dari penulisan dokumen ini diantaranya adalah sebagai berikut.

1. Memberikan gambaran alternatif desain yang dapat digunakan dalam mengembangkan rancang bangun *Fleet Monitoring dan Control System*,
2. Menjabarkan justifikasi terhadap desain yang dipilih untuk diimplementasikan dalam rancang bangun *Fleet Monitoring dan Control System*.

### **1.3 REFERENSI**

- [1] Simcom.EE. SIM908 Datasheet. <http://simcom.ee/modules/gsm-gprs-gps/sim908>. Diakses pada 24 November 2016 pukul 15.00.
- [2] Simcom.EE. SIM900 Datasheet. <http://simcom.ee/modules/gsm-gprs/sim900>. Diakses pada 24 November 2016 pukul 15.03.
- [3] ScalAgent. 2015. Benchmark of MQTT Servers.
- [4] Isode. M-Link & XMPP Performance Measurements over Satcom and Constrained IP Networks. <http://www.isode.com/whitepapers/xmpp-performance-constrained.html>. Diakses pada 30 November 2016 pukul 19.07.
- [5] Microchip. MCP2515 Stand-Alone CAN Controller with SPI Interface Datasheet.
- [6] Fabbri G., dkk. "Development of an On-Board Unit for the Monitoring and Management of an Electrical Fleet". 2012. XXth International Conference on Electrical Machines.
- [7] Salceanu, Andrei, dkk. "Monitoring the Environment by Enhancing the Capabilities of a Fleet Tracking System". 2014. International Conference and Exposition on Electrical and Power Engineering (EPE).
- [8] Sveum P., dkk. "Wireless Monitoring of an Electric Fleet Hub". 2011. IEEE Vehicle Power and Propulsion Conference.
- [9] Sriborriux, Wiroon, dkk. "The Design of RFID Sensor Network for Bus Fleet Monitoring". 2008. 8th International Conference on ITS Telecommunications.

## 1.4 DAFTAR SINGKATAN

SINGKATAN	ARTI
GPS	<i>Global Positioning System</i>
FMCS	<i>Fleet Monitoring and Control System</i>
ECU	<i>Engine Control Unit</i>
GUI	<i>Graphical User Interface</i>
CAN	<i>Controller Area Network</i>
UART	<i>Universal Asynchronous Receiver Transmitter</i>

## 2 DESKRIPSI UMUM

### 2.1 DEFINISI, FUNGSI DAN SPESIFIKASI

*Guided bus* merupakan bus yang beroperasi didalam lajur khusus sehingga bebas macet. Pada jalur *guided bus*, terdapat dinding pada sisi kiri dan kanan bus yang mengarahkan laju bus. *Guided bus* memiliki *guide wheel* atau roda pemandu. *Guided bus* dilengkapi dengan *Fleet Monitoring and Control System* (FMCS) yaitu sistem pemantauan dan kontrol *real-time*. Sistem ini akan mengirimkan data dari setiap armada ke *control station*. Data yang dikirimkan berupa posisi, suhu baterai dan level energi baterai yang kemudian akan ditampilkan pada GUI di *control station*.

Berikut adalah fungsi FMCS dan spesifikasi yang telah diturunkan dari fungsinya.

Tabel 1 Fungsi dan spesifikasi FMCS

Fungsi	Spesifikasi
Memantau posisi tiap armada.	Posisi yang dibaca oleh operator pada <i>control station</i> memiliki <i>error</i> maksimal sebesar 6 meter.
Memantau rpm kendaraan, kondisi fault, dan level energi baterai.	Dapat membaca data rpm kendaraan, kondisi fault, dan level energi baterai sesuai spesifikasi ECU yang sudah terdapat di <i>guided bus</i> .
Mengirimkan data ke <i>control station</i> melalui server.	Data yang dikirimkan adalah: <ul style="list-style-type: none"><li>• Posisi</li><li>• rpm kendaraan</li><li>• Kondisi fault</li><li>• Level energi baterai</li></ul> Data yang dikirim harus dapat dibaca pada GUI di <i>control station</i> maksimal setiap 0.8 detik.
Menampilkan data-data armada pada GUI di <i>control station</i> .	GUI pada <i>control station</i> menampilkan: <ul style="list-style-type: none"><li>• Posisi tiap armada</li><li>• rpm tiap armada</li></ul>

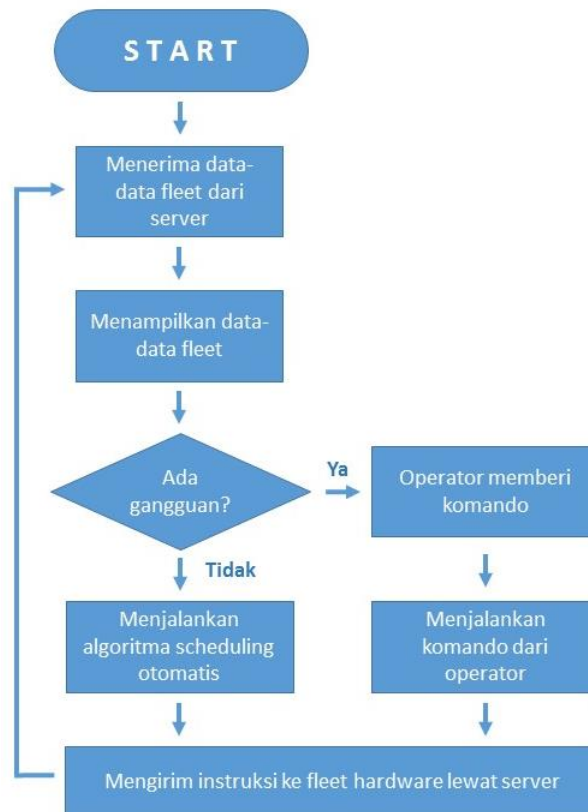
	<ul style="list-style-type: none"> <li>• Kondisi fault tiap armada</li> <li>• Level energi baterai tiap armada</li> </ul> <p>GUI ini akan dipantau dan dikendalikan oleh operator di <i>control station</i>.</p>
<i>Control station</i> dapat melakukan penjadwalan.	Ada algoritma yang dapat mengatur penjadwalan operasi tiap armada.
Mengirimkan perintah dari <i>control station</i> untuk pengemudi melalui server.	<i>Control station</i> dapat mengirimkan perintah operasional ke armada setiap terjadi perubahan perintah.
Data yang dikirim dari armada ke <i>control station</i> dan dari <i>control station</i> ke armada aman.	Data-data komunikasi antara armada dan <i>control station</i> hanya dapat diakses oleh armada yang bersangkutan dan operator pada <i>control station</i> .

Selain fungsi dan spesifikasi di atas, terdapat dua fitur tambahan yang akan diimplementasikan dalam FMCS ini. Fitur-fitur tersebut adalah *control station* dapat mengirimkan pesan ke tiap armada dan GUI yang mudah digunakan.

Untuk mempermudah operator melakukan pengawasan dan penjadwalan, GUI akan dibuat mudah untuk digunakan dan *user-friendly*. Pada GUI akan ditampilkan posisi armada, tidak hanya pada peta geografis, tetapi juga pada peta diagramatis, yaitu peta yang sudah dibentuk sedemikian rupa untuk menunjukkan koridor-koridor rute armada.

## 2.2 FLOWCHART SISTEM

### 2.2.1 Flowchart Algoritma pada GUI



Gambar 1 Flowchart algoritma GUI

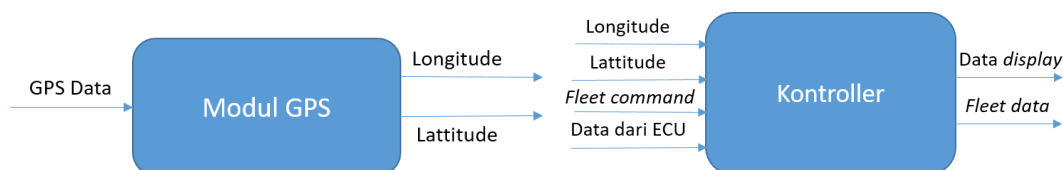
Flowchart diatas menggambarkan proses yang terjadi dalam GUI. Pertama-tama setelah dijalankan GUI akan selalu menerima data fleet dari server kemudian menampilkannya. Apabila tidak ada gangguan maka akan dijalankan instruksi dari algoritma scheduling otomatis, sementara jika terjadi gangguan maka operator akan memberikan instruksi secara manual. Semua instruksi dari algoritma utama kemudian diteruskan ke ECU *monitoring*.

## 2.3 DIAGRAM BLOK INPUT-OUTPUT

Pada bagian ini akan dijelaskan blok *input-output* setiap modul yang akan digunakan. Secara garis besar, modul yang akan digunakan dapat dibagi menjadi dua, yaitu modul *hardware* dan modul GUI.

### 2.3.1 Hardware

Dari sisi *hardware*, terdapat empat modul utama yang akan digunakan dalam FMCS ini, yaitu modul GPS, modul komunikasi, kontroller dan *display*.

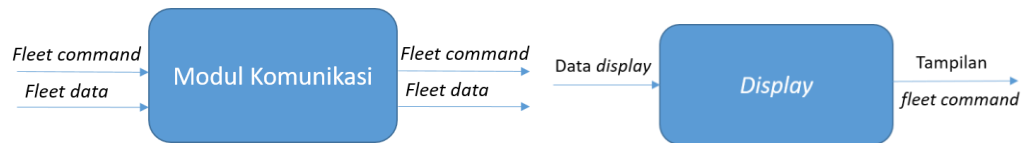


Gambar 2 Diagram modul GPS dan kontroller



Pada modul GPS, GPS data bertindak sebagai input untuk kemudian diproses menjadi data *longitude* dan *latitude* armada. Kedua data ini kemudian masuk ke kontroller bersama dengan data dari ECU (energi dan suhu baterai) untuk kemudian diproses menjadi *fleet data*.

Selain untuk menghasilkan *fleet data*, kontroller juga digunakan untuk memproses data *fleet command* dari *control station*. *Fleet command* yang masuk kemudian diproses menjadi data *display* untuk masuk modul *display*.



**Gambar 3 Diagram modul komunikasi dan display**

Modul komunikasi digunakan untuk proses pengiriman data *control station* dan armada. *Input* dari modul ini adalah *fleet command* dan *fleet data* yang siap dikirim, sedangkan *output*-nya juga *fleet command* dan *fleet data*.

Modul *display* digunakan untuk menampilkan *fleet command* dari *control station*. *Input* dari modul ini adalah data *display* dari controller untuk kemudian diproses menjadi *output* berupa tampilan *fleet command*.

### 2.3.2 Graphical User Interface di Control Station

Pada GUI di control station, terdapat tiga fungsi utama yang digunakan untuk menjalankan FMCS ini, yaitu algoritma utama, display data & interface, serta algoritma penjadwalan.



**Gambar 4 Diagram blok algoritma utama**

Algoritma utama menerima 3 jenis data. Yang pertama *fleet data and location* dari server, kemudian *scheduling command* dari algoritma penjadwalan, serta *fleet command* dari operator. Semua data ini diolah dalam algoritma utama dan dikeluarkan dalam bentuk *fleet data analysis* dan *fleet data and location* yang akan ditampilkan pada display data, serta *fleet command* yang akan diberikan kepada masing-masing hardware lewat server.



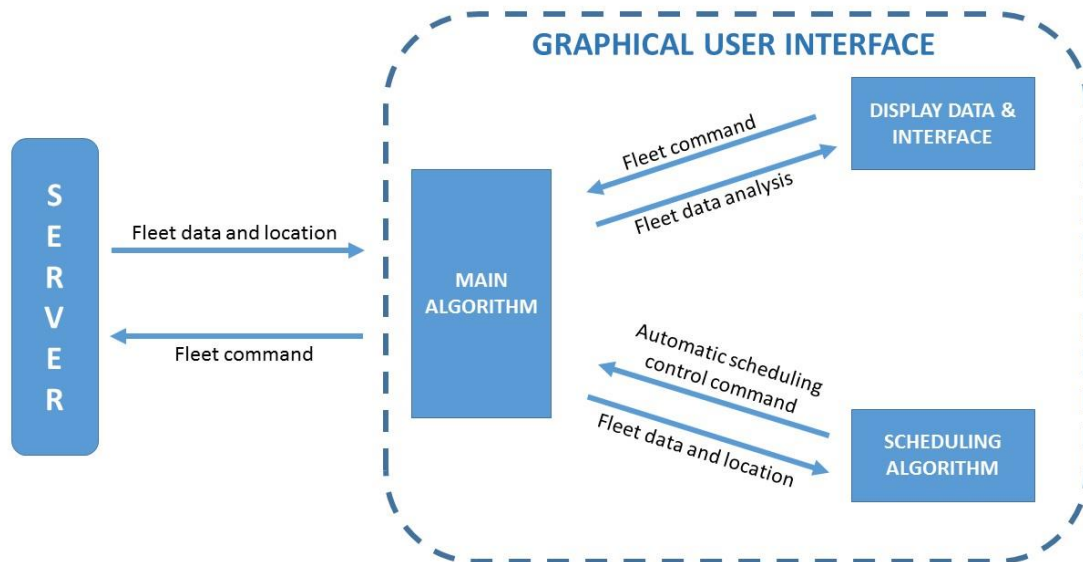
**Gambar 5 Diagram blok interface**

*Display Data & Interface* bertugas menyampaikan *fleet command* dari operator ke algoritma utama dan menampilkan *fleet data analysis* yang telah diolah oleh algoritma utama.



**Gambar 6 Diagram blok algoritma penjadwalan**

*Scheduling algorithm* akan menerima *fleet data and location* dari server dan bertugas mengirimkan *automatic scheduling control command* ke algoritma utama.



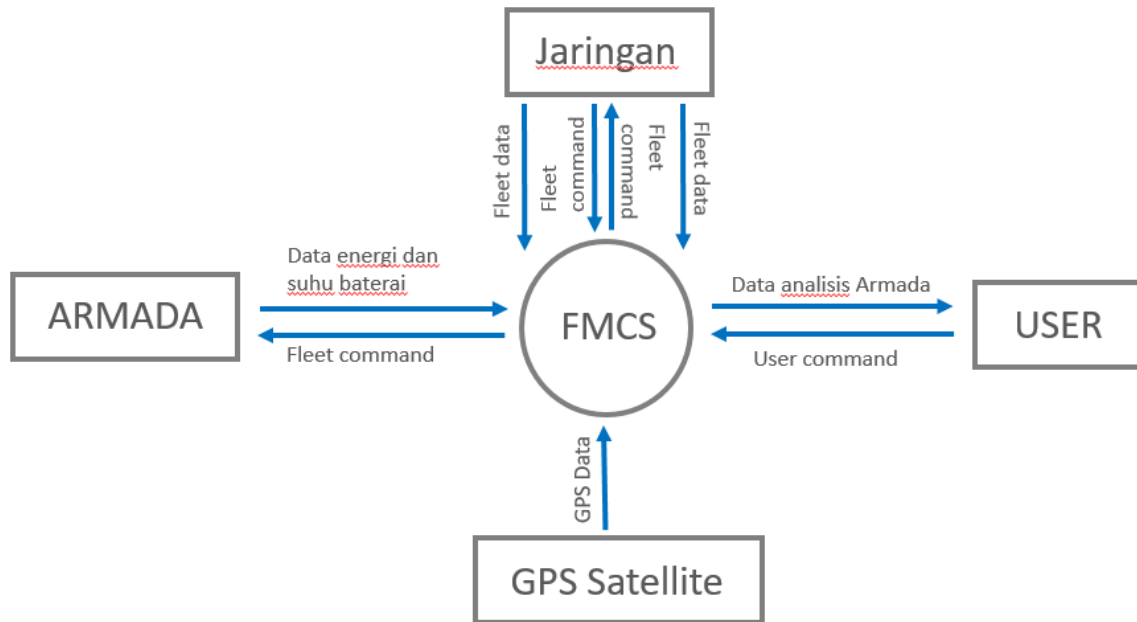
**Gambar 7 Diagram blok keseluruhan sistem GUI**

Secara keseluruhan, GUI menerima data berupa data fleet dan lokasi dari masing-masing fleet melalui server dan menyampaikannya ke algoritma penjadwalan. Oleh algoritma penjadwalan, data fleet dan lokasi diolah menjadi control command otomatis yang dikembalikan ke algoritma utama.

Jika terjadi kondisi darurat, operator dapat memberi komando melalui interface yang kemudian disampaikan ke algoritma utama. Perlu diketahui prioritas komando operator lebih tinggi dari algoritma scheduling, sehingga komando operator akan menginterupsi komando dari algoritma scheduling. Pada kondisi normal, algoritma utama akan terus menjalankan komando dari algoritma penjadwalan.

## 2.4 DATA FLOW DIAGRAM

### 2.4.1 DFD Level 0

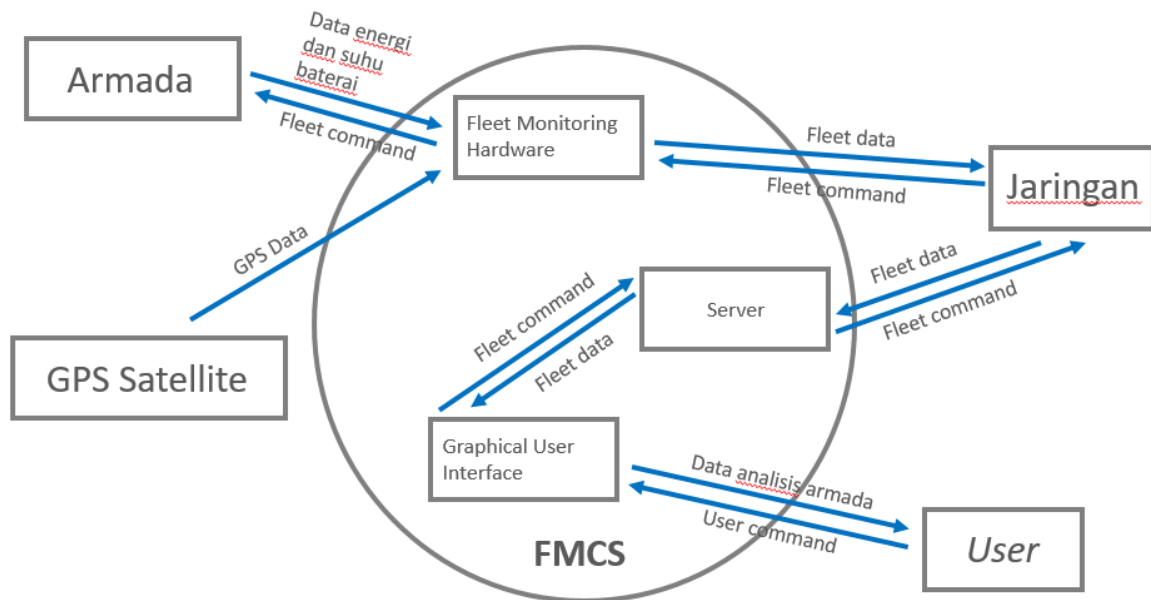


Gambar 8 DFD level 0

DFD level 0 menggambarkan sistem secara keseluruhan, batasan serta komponen yang terlibat dalam FMCS. Pada sistem ini terdapat 4 bagian yang terlibat, yaitu armada, *user*, *cloud* dan satelit GPS. *User* yang dimaksud dalam hal ini adalah pihak yang menggunakan FMCS.

Sistem ini menerima *input* dari tiga komponen, yaitu armada, satelit GPS dan *user*. Dari armada, sistem menerima *input* berupa energi dan suhu baterai. Dari satelit GPS, sistem menerima *input* berupa data GPS lokasi armada. Dari *user*, sistem menerima *input* berupa *fleet command*.

### 2.4.2 DFD level 1

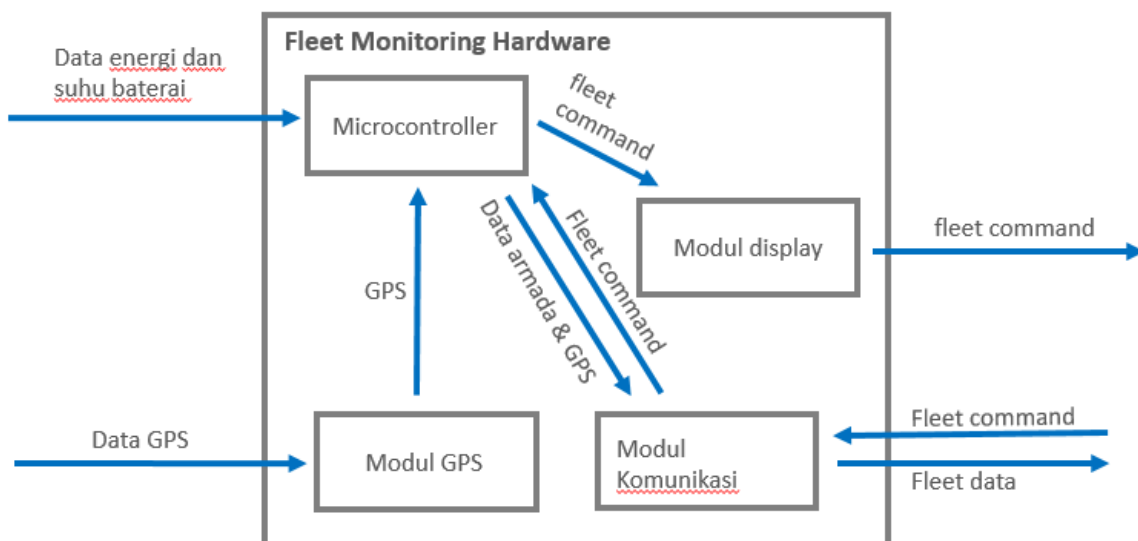


Gambar 9 DFD level 1

DFD level 1 menggambarkan sistem FMCS. Sistem FMCS terdiri dari tiga bagian, yaitu server, GUI dan ECU monitoring. *User* akan berinteraksi dengan ECU monitoring melalui GUI. Perintah yang diberikan *user* akan diteruskan ke server sampai pada ECU monitoring. ECU monitoring akan menampilkan perintah *user* untuk armada, setelah itu ECU monitoring akan merekam data armada. Terakhir, data dikirim ke GUI melalui server sehingga *user* dapat memantau keadaan *fleet*.

### 2.4.3 DFD level 2

#### 2.4.3.1 ECU monitoring



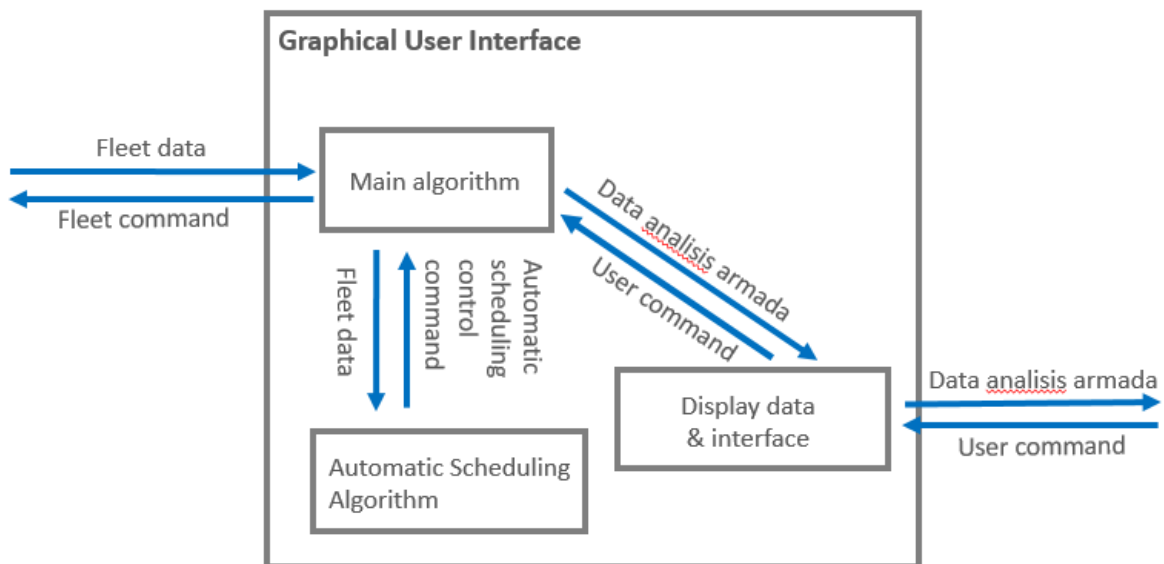
Gambar 10 DFD level 2 - hardware

ECU *monitoring* terdiri dari empat bagian, yaitu kontroller, modul *display*, modul GPS dan modul komunikasi.

Data energi baterai, suhu baterai (dari ECU) dan data dari GPS pertama-tama masuk ke dalam kontroller. Kontroller kemudian memproses data-data ini menjadi *fleet data*, untuk kemudian diteruskan ke modul komunikasi. Modul komunikasi kemudian mengirimkan *fleet data* tersebut ke server.

Untuk alur sebaliknya, *fleet command* dari *control station* masuk ke kontroller melalui modul komunikasi. *fleet command* ini kemudian diteruskan ke modul display untuk ditampilkan agar dapat terlihat oleh pengendara armada.

#### 2.4.3.2 GUI



Gambar 11 DFD level 2 - GUI

GUI merupakan media *user* untuk berinteraksi dengan data data yang ada, serta media bagi *user* dalam memberi perintah pada armada. GUI menerima masukan *fleet data* dari server untuk diolah dan ditampilkan. Selain melihat *fleet data* yang dikirimkan, *user* juga dapat memberikan perintah ke armada melalui GUI. Apabila tidak ada perintah dari *user*, maka *default*-nya perintah berasal dari algoritma penjadwalan otomatis yang juga merupakan hasil olahan dari data posisi armada saat ini.

### 2.4.3.3 Server



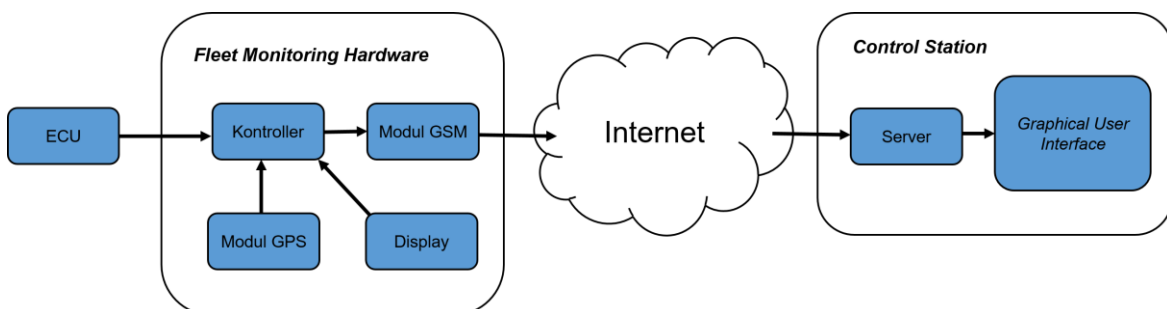
Gambar 12 DFD level 2 - server

Server adalah suatu sistem komputer yang menyediakan layanan untuk *client* dalam suatu jaringan komputer. Server dapat mengontrol akses dan sumber daya data yang berada dalam jaringannya. Pada sistem ini, server berfungsi untuk meneruskan dan menyampaikan data antara ECU *monitoring* dengan GUI pada *control station*.

## 3 ARSITEKTUR SISTEM

Pada bagian ini akan dijelaskan mengenai pilihan arsitektur yang dapat digunakan untuk memenuhi spesifikasi sistem yang diinginkan. Terdapat tiga pilihan arsitektur. Perbedaan utama dari ketiga pilihan ini ada pada letak server dan metode komunikasi yang digunakan. Berikut penjelasan untuk setiap pilihan arsitektur.

### 3.1 PILIHAN ARSITEKTUR PERTAMA

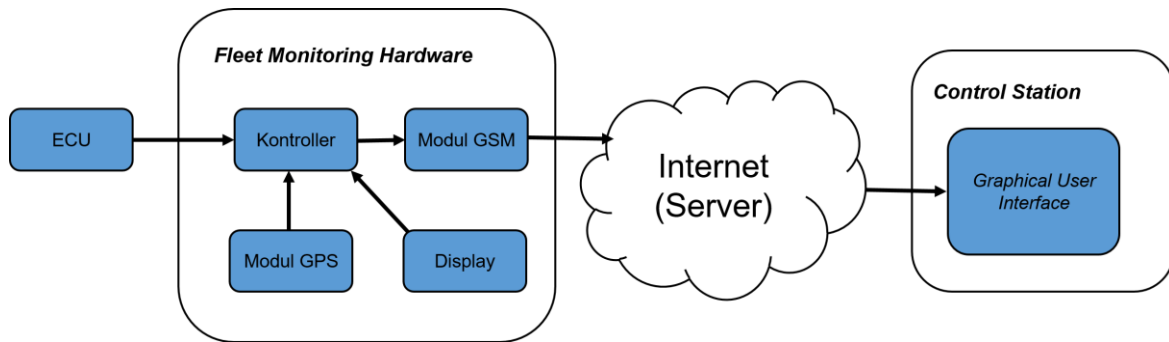


Gambar 13 Pilihan arsitektur pertama

Pilihan arsitektur pertama menggunakan modul GSM sebagai modul komunikasi. Pada arsitektur ini, server yang digunakan adalah server pribadi yang diletakkan di *control station*. Server ini menggunakan komputer sebagai *dedicated server*. Pada *control station* juga terdapat GUI yang digunakan untuk kendali dan *monitoring* armada.

Pada arsitektur ini, data yang berasal dari ECU dan modul GPS diterima oleh kontroller dan diolah. Setelah itu, data ini dikirim ke server menggunakan modul GSM melalui internet. Kemudian GUI akan mengambil data dari server melalui jaringan lokal. Untuk arah data sebaliknya, perintah dari GUI dikirim ke server melalui jaringan lokal. Kemudian, perintah ini diteruskan oleh server ke modul GSM pada ECU *monitoring* melalui internet. Perintah ini akan diolah oleh kontroller untuk kemudian memberikan tanda pada pengemudi armada.

### 3.2 PILIHAN ARSITEKTUR KEDUA

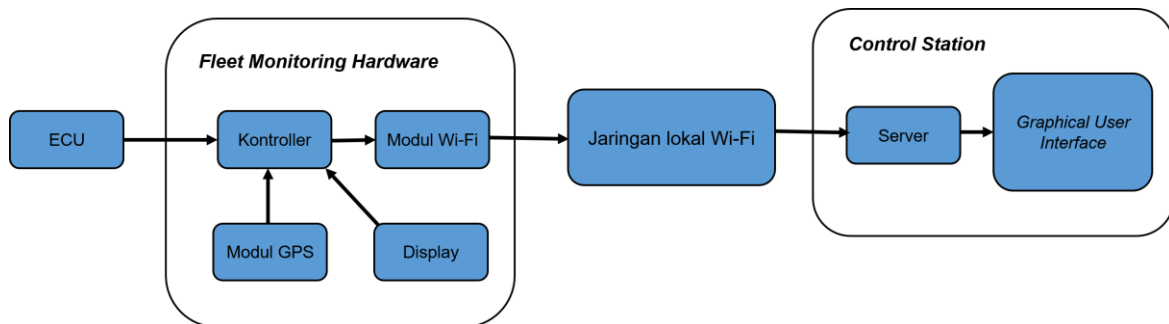


Gambar 14 Pilihan arsitektur kedua

Pilihan arsitektur kedua menggunakan modul GSM sebagai modul komunikasi. Pada arsitektur ini, server yang digunakan adalah server dari luar. Pengadaan server ini akan melibatkan pihak ketiga sebagai layanan penyedia server. Karena server yang digunakan terletak di luar *control station*, GUI yang digunakan untuk kendali dan *monitoring* armada terpisah dari servernya.

Pada arsitektur ini, data yang berasal dari ECU dan modul GPS diterima oleh kontroller dan diolah. Setelah itu, data ini dikirim ke server menggunakan modul GSM melalui internet. Kemudian GUI akan mengambil data dari server melalui internet. Untuk arah data sebaliknya, perintah dari GUI dikirim ke server melalui internet. Kemudian, perintah ini diteruskan oleh server ke modul GSM pada ECU *monitoring* melalui internet. Perintah ini akan diolah oleh kontroller untuk kemudian memberikan tanda pada pengemudi armada.

### 3.3 PILIHAN ARSITEKTUR KETIGA



Gambar 15 Pilihan arsitektur ketiga

Pilihan arsitektur pertama menggunakan modul Wi-Fi sebagai modul komunikasi. Pada arsitektur ini, server yang digunakan adalah server pribadi yang diletakkan di *control station*. Server ini menggunakan komputer sebagai *dedicated server*. Pada *control station* juga terdapat GUI yang digunakan untuk kendali dan *monitoring* armada.

Sepanjang jalur *guided bus* akan dipasang serangkaian modul wi-fi yang berfungsi sebagai *access point* yang saling terkoneksi satu sama lainnya. Serangkaian modul wi-fi ini berujung di *control station*, sehingga keseluruhan jaringan dari sepanjang jalur *guided bus* dan *control station* menjadi sebuah jaringan lokal yang besar. Pada ECU *monitoring* terdapat modul wi-fi yang berguna untuk menghubungkan armada dengan serangkaian modul wi-fi di sepanjang jalur *guided bus*.

Pada arsitektur ini, data yang berasal dari ECU dan modul GPS diterima oleh kontroller dan diolah. Setelah itu, data ini dikirim ke server menggunakan modul wi-fi melalui jaringan lokal wi-fi. Kemudian GUI akan mengambil data dari server melalui jaringan lokal. Untuk

arah data sebaliknya, perintah dari GUI dikirim ke server melalui jaringan lokal. Kemudian, perintah ini diteruskan oleh server ke modul GSM pada ECU *monitoring* melalui jaringan lokal wi-fi. Perintah ini akan diolah oleh controller untuk kemudian memberikan tanda pada pengemudi armada.

### 3.4 ARSITEKTUR SISTEM KESELURUHAN

Pada bagian ini akan dilakukan perbandingan terhadap ketiga pilihan arsitektur yang sudah dipaparkan pada bagian sebelumnya, Berikut adalah tabel untuk membandingkan ketiga pilihan arsitektur tersebut.

**Tabel 2 Perbandingan pilihan arsitektur**

Arsitektur	Pilihan Pertama	Pilihan Kedua	Pilihan ketiga
<b>Maintenance</b>	Posisi server dekat dengan operator	Posisi server tidak dekat dengan operator, butuh pihak ketiga sebagai penyedia layanan.	Posisi server dekat dengan operator
<b>Biaya implementasi dan kemudahan instalasi</b>	Jaringan internet memanfaatkan tower BTS (sudah ada dari operator seluler)	Jaringan internet memanfaatkan tower BTS (sudah ada dari operator seluler)	Harus membuat jaringan Wi-Fi sendiri disepanjang trayek.

Dengan membandingkan faktor-faktor tersebut, maka arsitektur yang dipilih adalah pilihan arsitektur pertama dengan modul komunikasi GSM dan server pribadi dalam *control station*.

Kelebihan dari pilihan arsitektur ini dibandingkan dengan pilihan arsitektur dengan server luar adalah server pribadi milik *control station* lebih mudah untuk dijaga performanya karena dekat dengan operator/teknisi. Selain itu, server pribadi juga mengurangi ketergantungan dengan pihak lain.

Kelebihan dari pilihan arsitektur ini dibandingkan dengan pilihan arsitektur dengan jaringan wi-fi lokal adalah biaya implementasi yang lebih murah, juga kemudahan instalasi sistem dibandingkan dengan memasang sistem jaringan wi-fi lokal.

## 4 DESAIN

Pada bagian ini akan dijelaskan mengenai desain sub-sistem yang diperlukan untuk menjalankan *fleet monitoring and control system*. Desain ini akan merujuk pada pilihan arsitektur yang telah ditetapkan sebelumnya, yaitu dengan menggunakan modul komunikasi GSM dan server pribadi dalam *control station*. Desain ini dibagi menjadi empat bagian sub-sistem, yaitu desain *hardware*, GUI, server dan algoritma penjadwalan. Selain bagian sub-sistem, bagian ini juga akan menjelaskan desain sistem keseluruhan yang akan digunakan, serta desain *hardware in-the-loop simulation* untuk membantu verifikasi sistem.

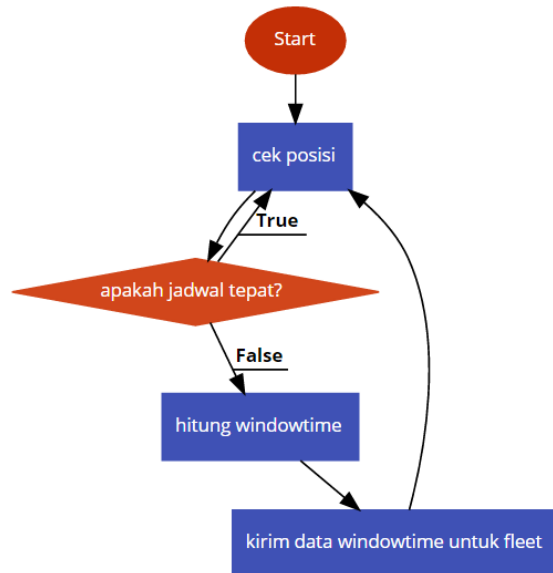
### 4.1 ALGORITMA PENJADWALAN

Algoritma penjadwalan berfungsi untuk mengatur kapan bus maju dan berhenti. Algoritma ini bertujuan agar persebaran bus merata pada jalur trayeknya sehingga penumpang tidak perlu menunggu bus terlalu lama. Algoritma ini memiliki beberapa pendekatan:

- Pendekatan jarak antar bus dengan mengatur *window time*



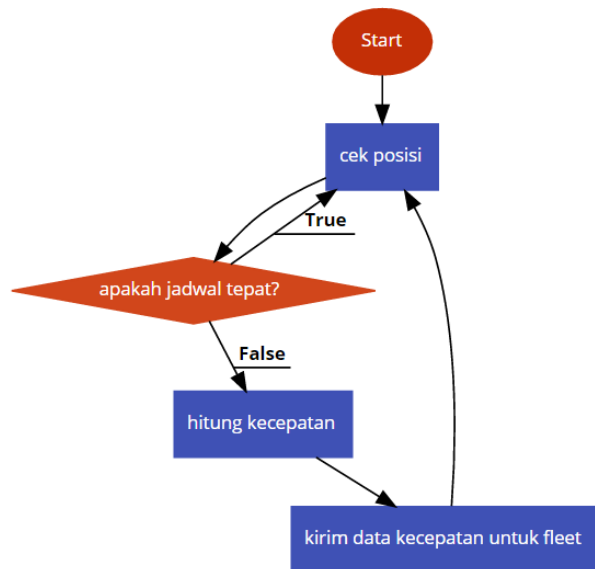
Pada algoritma ini jarak antar bus dijaga tetap sehingga bus tersebar merata. Jarak antar bus dapat dikontrol dengan mengatur waktu berhenti dan berangkat setiap bus pada masing masing halte (*window time*). Apabila jarak antar dua bus berkurang maka *window time* bus belakang akan bertambah sedangkan *window time* pada bus depan akan berkurang sehingga jarak kedua bus ini kembali pada jarak seharusnya. Sebaliknya apabila terdapat 2 bus yang saling berjauhan maka *window time* bus belakang akan berkurang sedangkan *window time* pada bus depan akan bertambah sehingga jarak kedua bus ini kembali pada jarak seharusnya.



**Gambar 16** Algoritma penjadwalan dengan mengatur *window time*

- Pendekatan jarak antar bus dengan mengatur kecepatan

Pada pendekatan ini jarak antar bus dijaga tetap dengan mengatur kecepatan bus dan membuat *window time* yang konstan. Jika terdapat 2 bus, misal bus A dan bus B, yang saling berjauhan maka kecepatan salah bus A dipercepat dengan memperhitungkan jarak bus tersebut dengan bus lainnya, apabila menaikkan kecepatan bus A berdampak menjauhkan bus A dengan bus di belakangnya maka bus B harus menurunkan kecepatan sedangkan bus A mempertahankan kecepatannya.



**Gambar 17 Algoritma penjadwalan dengan mengatur kecepatan armada**

Dalam pemilihan algoritma ini digunakan beberapa asumsi yaitu:

- jumlah persebaran penumpang sama disetiap haltenya, dan
- bus berjalan pada jalur khusus yang bebas macet.

**Tabel 3 Perbandingan algoritma penjadwalan**

keterangan	Pendekatan jarak antar bus dengan mengatur kecepatan		Pendekatan jarak antar bus dengan mengatur <i>window time</i>	
	+	-	+	-
konsumen	Tidak membuat penumpang menunggu di halte			Membuat penumpang menunggu di halte
implementasi		Jika dilakukan manual susah untuk menyesuaikan kecepatan	Mudah impementasi oleh pengemudi	
Kehandalan sistem	Window untuk menyesuaikan scheduling banyak			Window untuk menyesuaikan scheduling sedikit

Terdapat beberapa poin yang menjadi pertimbangan yaitu konsumen, konsumen lebih suka jika kendaraan melaju walaupun dengan kecepatan rendah dibandingkan dengan jika kendaraan diam. Pada implementasinya algoritma dengan pendekatan jarak antar bus dengan mengatur *window time* lebih mudah di implementasikan karena pengemudi hanya mengatur kapan bus berhenti dan melaju. Sedangkan pada algoritma pendekatan jarak antar bus dengan mengatur kecepatan lebih sulit di implementasikan karena pengemudi harus

menyesuaikan kecepatan setiap saat dengan kecepatan yang diperintahkan oleh *control station*. poin yang menjadi pertimbangan selanjutnya adalah kehandalan sistem, yang dimaksud dengan kehandalan sistem adalah kemampuan sistem untuk mengontrol armada apabila terjadi ketidaksesuaian penjadwalan dengan implementasi di lapangan. Algoritma dengan Pendekatan jarak antar bus dengan mengatur kecepatan memiliki lebih banyak kesempatan untuk menyesuaikan dengan jadwal seharusnya karena bus lebih banyak menghabiskan waktu dalam perjalanan dibandingkan dengan berhenti di halte.

## 4.2 DESAIN GRAPHICAL USER INTERFACE

*Graphical User Interface* (GUI) merupakan tampilan antarmuka dalam bentuk grafis sebagai media interaksi antar pengguna dengan sistem operasi. Secara garis besar, pada sistem ini GUI harus dapat mengerjakan tugas-tugas berikut.

1. Menampilkan posisi setiap armada dalam bentuk peta geografis dan diagramatis.
2. Menampilkan level baterai dari setiap armada.
3. Mengirim komando kepada masing-masing armada.
4. Pengaturan aplikasi.

### 4.2.1 Peta geografis

Peta merupakan gambaran permukaan bumi pada bidang dua dimensi dengan skala tertentu melalui suatu sistem proyeksi. Peta berguna untuk menunjukkan dan menggambarkan lokasi atau letak suatu kawasan, wilayah, atau objek geografis lainnya. Umumnya peta adalah peta geografis, yaitu peta yang sesuai dengan objek yang sesungguhnya.


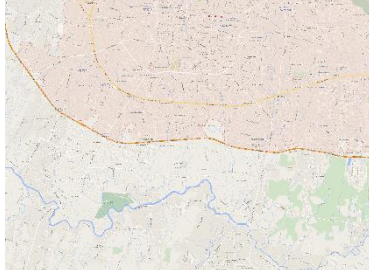


**Gambar 18 Peta geografis**

Pada GUI, peta geografis digunakan untuk menampilkan posisi dari masing-masing armada. Peta geografis sendiri bisa didapatkan dengan menggunakan beberapa metode, diantaranya dengan menggunakan *embedded map*, dan dengan membuat file gambar vector sendiri. Berikut merupakan tabel yang dapat digunakan untuk membandingkan *trade off* dari masing-masing metode :

**Tabel 4 Perbandingan metode mendapatkan peta geografis**

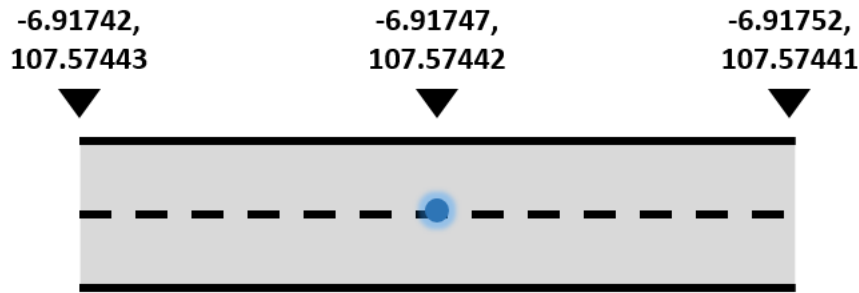
Jenis	<i>Embedded map</i>	File gambar vector
-------	---------------------	--------------------

Jangkauan lokasi	Jangkauannya lebih luas (hingga seluruh dunia)	Jangkauannya terbatas (hanya lokasi yang terdapat dalam gambar)
Resolusi gambar	<i>Zoom level</i> maksimal	<i>Zoom level</i> terbatas pada saat pembuatan peta
Kecepatan loading gambar	Lebih lama (karena dari internet)	Lebih cepat (karena sudah tersimpan dalam komputer)
Keakuratan skala	Lebih akurat karena diambil langsung dari penyedia <i>source embedded map</i> .	Kurang akurat karena proses pembuatan gambar vector masih secara manual (menyatukan beberapa gambar vector dengan menggunakan <i>software image processing</i> ).
Kecepatan koneksi internet yang dibutuhkan	Tinggi, karena harus me- <i>load</i> gambar peta dari awal setiap kali dibutuhkan.	Tidak membutuhkan kecepatan yang tinggi karena file gambar peta sudah tersimpan dalam komputer.
Tampilan		

Berdasarkan tabel perbandingan yang telah dijabarkan diatas, diputuskan untuk menggunakan *Embedded map* karena lebih banyaknya *benefit* yang didapatkan dibandingkan kerugiannya. Adapun alasan-alasannya adalah sebagai berikut.

1. Jangkauan lokasi yang lebih luas.
2. Resolusi gambar yang lebih tinggi.
3. Gambar lebih akurat.

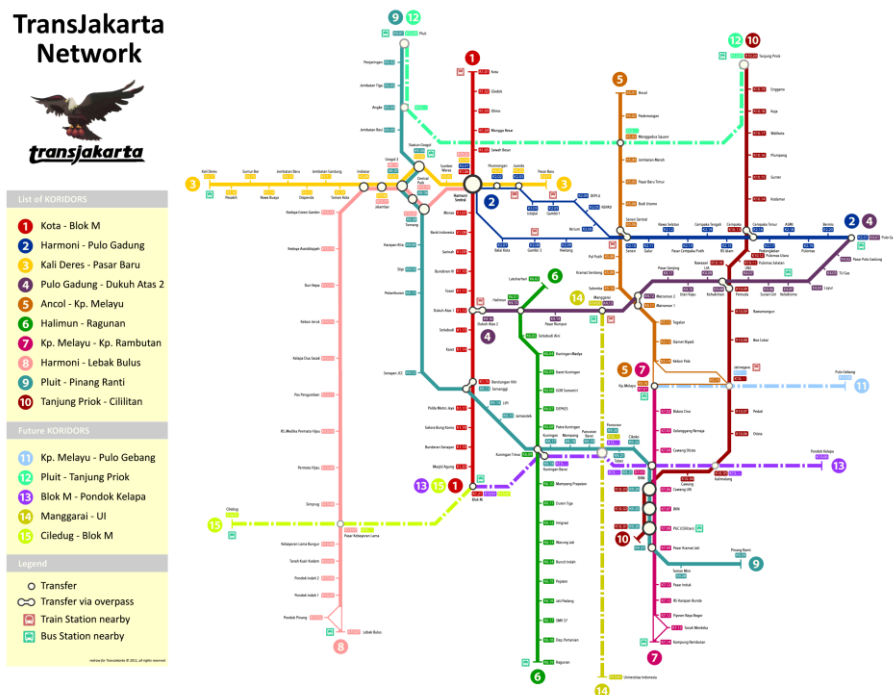
Trayek yang akan dibuat pada *prototype* adalah Jalan Soekarno Hatta sepanjang 18 kilometer. Dengan mempertimbangkan keakuratan modul komunikasi sebesar 6 meter, diputuskan untuk melakukan segmentasi setiap 12 meter sekali. Yang dimaksud disini adalah, untuk setiap pergerakan bus sepanjang 12 meter, hanya akan diwakili oleh satu koordinat. Hal ini dirancang untuk meminimalisir galat dari pengiriman data modul komunikasi.



Gambar 19 Contoh segmentasi

#### 4.2.2 Peta diagramatis

Peta diagramatis merupakan gambaran permukaan bumi pada bidang dua dimensi yang menggunakan abstraksi dari penggambaran objek yang sesungguhnya. Peta diagramatis bekerja dengan konsep menghilangkan informasi-informasi yang tidak dibutuhkan dalam menampilkan gambar lokasi.

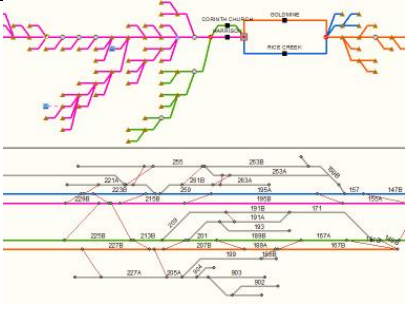



Gambar 20 Contoh peta diagramatis (Transjakarta)

Peta diagramatis bisa didapat dengan memetakan koordinat asli dari setiap objek pada peta geografis ke beberapa garis lurus dengan menggunakan perhitungan yang tepat. Peta diagramatis biasanya digunakan untuk *monitoring* angkutan masal agar lebih mudah dilihat oleh operator.

Peta diagramatis bisa didapatkan dengan menggunakan beberapa metode, diantaranya dengan menggunakan software ArcGIS, atau membuat file gambar vector dengan menggunakan perhitungan sendiri. Berikut merupakan tabel yang dapat digunakan untuk membandingkan *trade off* dari masing-masing metode :

**Tabel 5 Perbandingan metode mendapatkan peta diagramatis**

Jenis	Maps creator software	File gambar vector
Keakuratan	Lebih akurat karena kalkulasi dilakukan oleh prosesor.	Masih kemungkinan terjadi <i>miss</i> karena menggunakan komputasi manual dan proyeksi perkiraan.
Tampilan grafis		

Berdasarkan tabel perbandingan yang telah dijabarkan diatas, diputuskan untuk menggunakan *Maps creator software* karena lebih akurat.

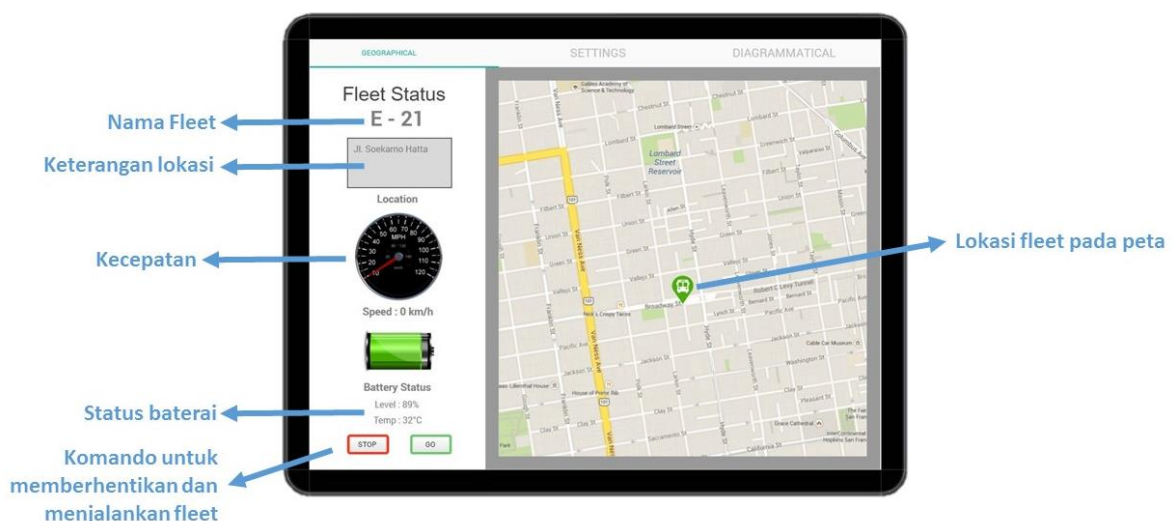
#### 4.2.3 Pengiriman komando

Untuk memenuhi spesifikasi dan fungsi sistem yang telah ditentukan, GUI dirancang untuk bisa menyampaikan komando dari operator di *control station* ke server. Dari server, komando akan dikirimkan ke *hardware* di masing-masing armada untuk dibaca oleh *driver*. Komando yang dapat diberikan *control station* diantaranya adalah :

1. Instruksi untuk memberhentikan armada
2. Instruksi untuk menjalankan armada

#### 4.2.4 Layout dan Platform GUI

Untuk melaksanakan tugas-tugas diatas, desain awal GUI yang telah dirancang secara garis besar adalah sebagai berikut :

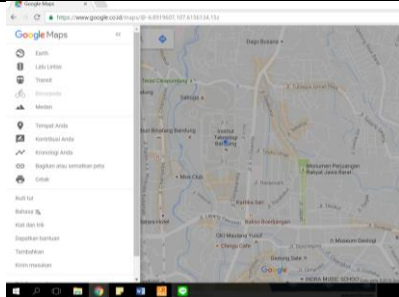
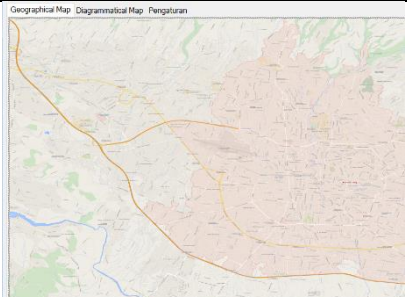


**Gambar 21 Desain GUI**



Untuk mengimplementasikan desain seperti diatas, terdapat dua alternatif *platform* GUI yang dapat dibuat. Berikut merupakan tabel yang dapat digunakan untuk membandingkan *trade off* dari masing-masing desain.

**Tabel 6 Perbandingan *platform* GUI**

Spesifikasi	Website	Windows Application
Display	Desain lebih terbatas karena hanya dapat mengikuti template yang telah disediakan oleh penyedia layanan web.	Desain dapat lebih variatif karena <i>asset</i> tidak terbatas seperti pada website.
Koneksi Internet	Cenderung dibutuhkan kecepatan akses internet yang lebih tinggi karena harus mengolah semua proses pada halaman web.	Dapat berjalan pada kecepatan akses normal karena hanya dibutuhkan untuk mengirim data dari server.
Cost	Biaya untuk menyewa server	Tidak perlu menyewa server
Fleksibilitas	Tidak perlu menginstall file.exe	Harus menginstall/memiliki file.exe
Contoh tampilan		

## 4.3 DESAIN SERVER DAN PROTOKOL KOMUNIKASI

### 4.3.1 Protokol Komunikasi

Untuk menghubungkan antara hardware yang terletak pada armada dengan *control station*, dibutuhkan jaringan internet sehingga dibutuhkan suatu protokol komunikasi tertentu agar *ECU monitoring* dan *control station* dapat berkomunikasi melalui internet.

Terdapat beberapa protokol komunikasi yang menghubungkan *device* (pada armada) dan server, yaitu MQTT dan XMPP. Kedua protokol ini memiliki kelebihan dan kekurangan seperti ditunjukkan pada tabel berikut.

**Tabel 7 Perbandingan protokol komunikasi**

Jenis	MQTT	XMPP
Transport	TCP	TCP
Messaging	Publish/Subscribe	Request/Response
SECURITY	medium	high
Protocol (tanpa enkripsi)	28 bytes	491 bytes

<b>Protocol (dengan enkripsi)</b>	68 bytes	308 bytes
<b>QoS</b>	Ada QoS	Tidak ada QoS
<b>tipe</b>	Asynchronous	Synchronous
<b>Resources Utilization</b>	Rendah	Tinggi
<b>Latency</b>	Millisecond <sup>[3]</sup>	second <sup>[4]</sup>

Dari tabel di atas, MQTT menggunakan *resource utilization* yang lebih rendah dibandingkan XMPP sehingga MQTT tidak membutuhkan microcontroller yang memiliki clock tinggi. Selain itu, MQTT bersifat *asynchronous* yaitu server dan *device* tidak harus dalam keadaan siap untuk menerima pesan. Berbeda dengan XMPP yang menggunakan metoda *polling* dalam bertukar informasi antara *device* dan server. Jaringan GSM yang memungkinkan data hilang atau rusak pada proses komunikasi dengan server, MQTT memiliki protokol data yang lebih sedikit dibandingkan dengan XMPP sehingga MQTT lebih tahan terhadap jaringan yang tidak stabil. Selain itu MQTT didukung dengan fitur QoS sehingga apabila ada data yang hilang pada proses komunikasi dengan server, maka server akan meminta data yang hilang tersebut sampai data tersebut diterima oleh server. *Latency* yang dimiliki oleh XMPP berkisar pada orde second sedangkan *latency* yang dimiliki oleh MQTT berkisar pada orde millisecond. Spesifikasi dari FMCS mengharuskan hardware mengupdate data tiap 0.6 detik sehingga MQTT dipilih sebagai protokol yang digunakan untuk komunikasi antara fleet dengan control station.

### 4.3.2 Server

#### 4.3.2.1 CPU

Untuk menerima data dari fleet dibutuhkan server yang berada di control station bertugas mengirimkan data ke armada-armada juga menerima data-data dari armada tersebut.

Menurut hasil benchmark yang dilakukan oleh Scalagent<sup>[3]</sup> dengan kondisi sebagai berikut:

- Jumlah client sekitar 1000 client
- Terdapat 10000 message/s
- 1 subscriber

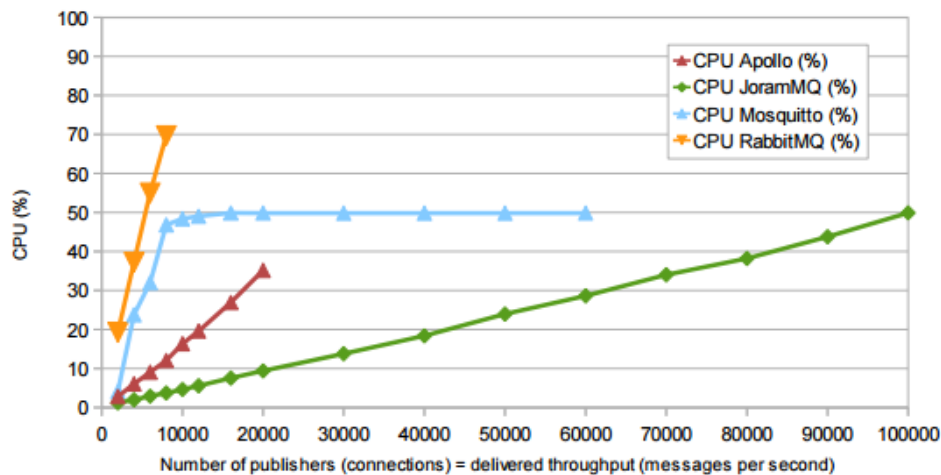
Digunakan server dengan spesifikasi sebagai berikut:

**Tabel 8 Perbandingan server**

Jenis	parameter
<b>CPU</b>	Intel Core 2 Duo CPU E8400 3.00GHz
<b>RAM</b>	4 GB
<b>HDD</b>	SATA 7200 RPM
<b>Network</b>	Gigabit switch

Hasil *benchmark* menunjukkan rata rata CPU usage adalah 50%. Seperti pada tabel dibawah.





**Gambar 22 Hasil benchmark CPU usage terhadap jumlah pesan**

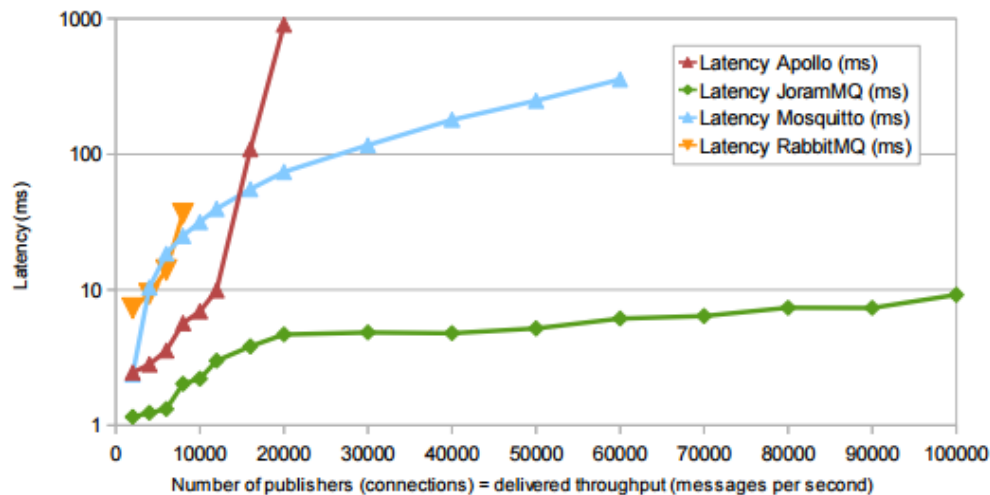
Kondisi ini dijadikan acuan dalam menentukan spesifikasi server yang digunakan dalam FMCS ini. Asumsi kondisi FMCS saat ber operasi adalah :

- Jumlah client sekitar 1000 client
- Terdapat maksimal 2000 message/s ( message setiap 0.6 detik tiap client )
- 1 subscriber

Jika melihat pada tabel hasil benchmark Scalagent, rata rata CPU usage saat menangani 2000 message per second adalah 10%. Pemakaian CPU usage server saat menangani 2000 message/s menjadi 1/5 kali pemakaian saat menangani 10000 message per second. Dengan demikian dapat disimpulkan spesifikasi server diatas sudah mencukupi untuk menjalankan FMCS.

#### 4.3.2.2 Message Broker

Message Broker merupakan program yang melanjutkan pesan dari pengirim ke penerima. Broker MQTT ini berfungsi sebagai penyalur pesan tanpa menampung pesan tersebut di server. Terdapat beberapa broker yang menyediakan server MQTT, baik gratis maupun berbayar. Pertimbangan dalam pemilihan broker ini antara lain adalah latensi server. Berikut grafik yang menunjukkan latensi beberapa broker yang tersedia pada jumlah klien yang terhubung ke server.



Gambar 23 Hasil *benchmark latency* terhadap jumlah pesan

Dari gambar di atas, dilihat kondisi pada klien sekitar 1000 dengan 10000 message/s dan 1 *subscriber*. Dari keempat server yang dites, dapat dilihat server Mosquitto memiliki latensi di bawah 10 ms pada kondisi tersebut. Oleh Karena pertimbangan latensi dan biaya server, dipilih server Mosquitto karena latensi yang masuk ke dalam spesifikasi dan merupakan server gratis tanpa biaya.

#### 4.4 DESAIN HARDWARE

Secara garis besar, *hardware* pada sistem ini akan digunakan untuk mengerjakan empat tugas, yaitu :

1. membaca data baterai dari ECU,
2. mendeteksi posisi armada,
3. menampilkan perintah dari *control station*, dan
4. mengirimkan data baterai dan posisi armada ke *control station*.

Untuk melakukan keempat tugas ini, diperlukan beberapa sub-modul berikut.

##### 4.4.1 Modul Akuisisi Data ECU

Pada kendaraan listrik, terdapat sebuah ECU utama yang mengumpulkan data-data parameter mobil listrik. Data-data parameter ini diperoleh dari banyak ECU yang memiliki tugas khusus, salah satunya merupakan ECU yang bertugas untuk memantau kondisi baterai. Seluruh ECU khusus ini berkomunikasi dua arah dengan ECU utama menggunakan protokol komunikasi *Controller Area Network* (CAN). Selain itu, pada setiap ECU terdapat *port* khusus untuk protokol komunikasi *Universal Asynchronous Receiver Transmitter* (UART).

ECU *monitoring* akan berkomunikasi dengan ECU lain pada *guided bus* melalui protokol komunikasi CAN. Oleh karena itu, digunakan sebuah modul CAN-Bus MCP2515 agar controller dapat berkomunikasi dengan ECU lainnya. Berikut tabel spesifikasi MCP2515<sup>[5]</sup>.

Tabel 9 Spesifikasi MCP2515

Parameter	Nilai
Version Supported	2.0B Active

<b>Tx Buffers</b>	3
<b>Rx Buffers</b>	2
<b>Masks</b>	2
<b>Interrupt Output</b>	1
<b>Filters</b>	6
<b>Temperature Range</b>	-40 to +125 °C
<b>Operating Voltage Range</b>	2.7 to 5.5 V

#### 4.4.2 Modul Komunikasi

Modul yang dibutuhkan berikutnya adalah modul komunikasi untuk menghubungkan ECU *monitoring* dengan *control station*. Komunikasi yang dipilih akan menggunakan jaringan GPRS.

Ada banyak modul komunikasi yang tersedia di pasaran. Modul-modul ini ada yang dijual dalam bentuk *chip*-nya saja, maupun dalam bentuk *shield* yang siap pakai. Untuk modul komunikasi yang berbentuk *chip*, diperlukan berbagai rangkaian tambahan agar modul tersebut bisa digunakan. Untuk *shield*, rangkaian tambahan ini sudah termasuk di dalam *shield* sehingga bersifat *plug-and-play*.

Terdapat dua jenis modul komunikasi GPRS dari SIMCOM yang terbaru pada saat dokumen ini ditulis. Berikut adalah tabel untuk membandingkan kedua jenis modul komunikasi GPRS tersebut.

Tabel 10 Spesifikasi modul komunikasi

Jenis	SIM908	SIM900
<b>Frekuensi Kerja</b>	Quad-Band 850/900/1800/1900 MHz	Quad-Band 850/900/1800/1900 MHz
<b>Power Consumption</b>	Class 4 (2 W @850/900 MHz) Class 1 (1 W @1800/1900 MHz)	Class 4 (2 W @850/900 MHz) Class 1 (1 W @1800/1900 MHz)
<b>Supply Voltage</b>	3.2-4.8 Volts	3.2-4.8 Volts
<b>Command</b>	GSM 07.07, 07.05 and SIMCOM enhanced AT Commands	GSM 07.07, 07.05 and SIMCOM enhanced AT Commands
<b>Operation Temperature</b>	-40 to 85 °C	-40 to 85 °C
<b>Data Transfer</b>	GPRS class 10: max. 85.6 kbps (downlink) CSD up to 14.4 kbps	GPRS class 10: max. 85.6 kbps (downlink) CSD USSD



#### 4.4.3 Modul GPS

Modul pelacak posisi yang digunakan adalah modul GPS. Karena menggunakan modul GPS, maka diperlukan sebuah antenna yang diletakkan di tempat yang cukup terbuka untuk melakukan *locking* GPS.

Terdapat banyak jenis dan merk modul GPS yang dijual di pasaran. Pada waktu pembuatan dokumen ini, terdapat dua merk modul GPS yang memiliki sedikit perbedaan di antara keduanya. Berikut tabel untuk membandingkan kedua merk modul GPS tersebut.

Tabel 11 Spesifikasi GPS

Jenis	Neo-6M	Neo-8M
<b>Input Voltage</b>	2.7-3.6 Volts	1.65-3.6 Volts
<b>Interface</b>	UART, USB, SPI and DDC	UART, USB, SPI and DDC
<b>Receiver Type</b>	50-channel u-blox 6 engine GPS L1 C/A code SBAS: WAAS, EGNOS, MSAS	72-channel u-blox M8 engine GPS L1C/A SBAS L1C/A QZSS L1C/A GLONASS L1OF BeiDou B1 Galileo E1B/C2
<b>Update Rate</b>	Up to 5 Hz	Up to 10 Hz (GPS), Up to 5 Hz (GPS & GLONASS)
<b>Accuracy</b>	2.5 meters	2.5 meters
<b>Start Time</b>	27 s cold start, 1 s hot start	27 s cold start (GPS & GLONASS), 30 s cold start (GPS), 1 s hot start
<b>Supported Antenna</b>	Active and Passive	Active and Passive
<b>Operating Temperature</b>	-40 to 85 °C	40 to 105 °C

Dengan pertimbangan penggunaan di Indonesia, satelit GLONASS sangat berpengaruh terhadap presisi dan akurasi hasil pembacaan lokasi dengan GPS, maka modul GPS yang akan digunakan adalah GPS Neo-8M dengan antenna aktif.

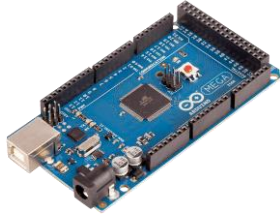

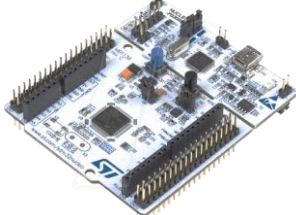
#### 4.4.4 Kontroller

*Hardware* pada sistem ini harus mampu mengolah data-data yang diperoleh dari ECU dan modul GPS dan berkomunikasi dengan *control station*. Oleh karena itu, dibutuhkan sebuah sistem *embedded* untuk mengatur jalannya seluruh data.

Pada bagian ini akan dibandingkan beberapa pilihan modul sistem *embedded* yang dapat digunakan untuk mengatur aliran data dari GPS dan ECU dan mengatur komunikasi dengan *control station*. Berikut adalah tabel untuk membandingkan pilihan sistem *embedded* yang dapat digunakan.

**Tabel 12** Spesifikasi sistem *embedded*

Jenis	Arduino Mega 2560	Raspberry Pi B	Nucleo-F401RE
<b>Fitur</b>	54 GPIO (4 UART termasuk soft-serial, 1 SPI, 1 I <sup>2</sup> C) 16 Analog input pins	40 GPIO (1 UART) 4 USB ports Full HDMI port Ethernet port 3.5mm analogue audio-video jack Camera serial interface (CSI) Display serial interface (DSI) Micro SD card slot Arduino Uno R-3 Connectivity	50 GPIO (3 I <sup>2</sup> C, 3 SPI, 4 UART) RTC Advanced-control Times 7 General Purpose Timers 2 Watchdog Timers 12-bit ADC with 16 channels
<b>Prosesor</b>	16 MHz ATmega 2560	1.2 GHz quad-core ARM Cortex-A53 1 GB RAM	84 MHz ARM 32-bit Cortex-M4 CPU with FPU
<b>Memori</b>	8 KB SRAM, 4 KB EEPROM, 256 KB FLASH	8 KB SRAM, 4 KB EEPROM, 1 MB SRAM, 1 MB EEPROM, 4 MB FLASH	512 KB FLASH, 96 KB SRAM
<b>Input Voltage</b>	7-12 Volts	4.75-5.25 Volts	USB VBUS or external source (3.3, 5 or 7-12 Volts)
<b>Operating Voltage</b>	5 Volts	3.3 Volts	3.3 Volts
<b>IDE</b>	Arduino (C++ based).	Programmed by C, Python, PHP, Javascript, etc.	Many, including IAR, Keil, GCC-based IDE, and online IDE.

<b>Dimensi</b>	101.52 mm x 53.3 mm	85.6 mm x 56 mm	80 mm x 69 mm x 20 mm
<b>Berat</b>	37 grams	45 grams	81 grams
<b>Harga</b>	Rp 150.000,-	Rp 700.000,-	Rp 250.000,-
<b>Bentuk</b>			

ECU *monitoring* membutuhkan setidaknya dua UART, sebuah I<sup>2</sup>C dan sebuah SPI. Pada sistem ini, kecepatan controller tidak terlalu dipentingkan karena tidak terdapat proses berat seperti pengolahan citra (hanya ada proses yang menggunakan *resource utilization* yang rendah). Dengan kedua pertimbangan ini, maka controller Arduino sudah cukup untuk memenuhi spesifikasi yang dibutuhkan.

#### 4.4.5 Display

Display pada ECU *monitoring* ini digunakan untuk menampilkan kondisi *hardware* dan menampilkan perintah dari *control station*. Perintah yang datang berupa perintah untuk mulai melaju dan perintah untuk berhenti (menunggu di halte). Untuk itu, ada beberapa pilihan untuk display pada *hardware*. Pilihan pertama adalah dengan menggunakan sebuah LCD *display*. Pilihan kedua adalah dengan menggunakan lampu-lampu indikator.

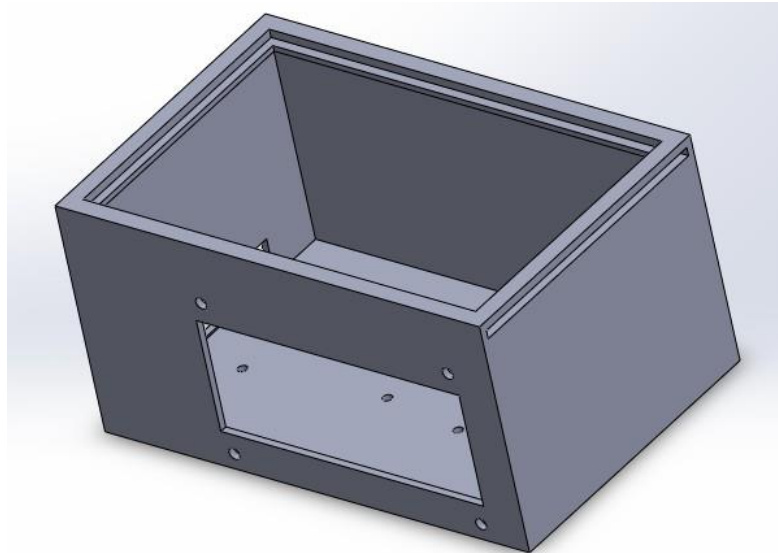
Karena pada bagian sebelumnya telah ditentukan desain algoritma penjadwalan dengan mengatur kecepatan armada, maka dibutuhkan suatu modul *display* yang dapat menampilkan kecepatan armada yang diperintahkan oleh *control station*. Salah satu modul *display* yang dapat digunakan adalah modul LCD *display* berukuran 20x4.

#### 4.4.6 Sumber Daya Listrik

Untuk memberikan daya pada seluruh komponen pada ECU *monitoring*, diperlukan sebuah sumber daya listrik DC dengan tegangan 5 volt atau berkisar antara 7-12 volt. Sumber daya ini dapat diperoleh dengan mengambil daya langsung dari sumber daya listrik pada *guided bus*.

#### 4.4.7 Desain Packaging

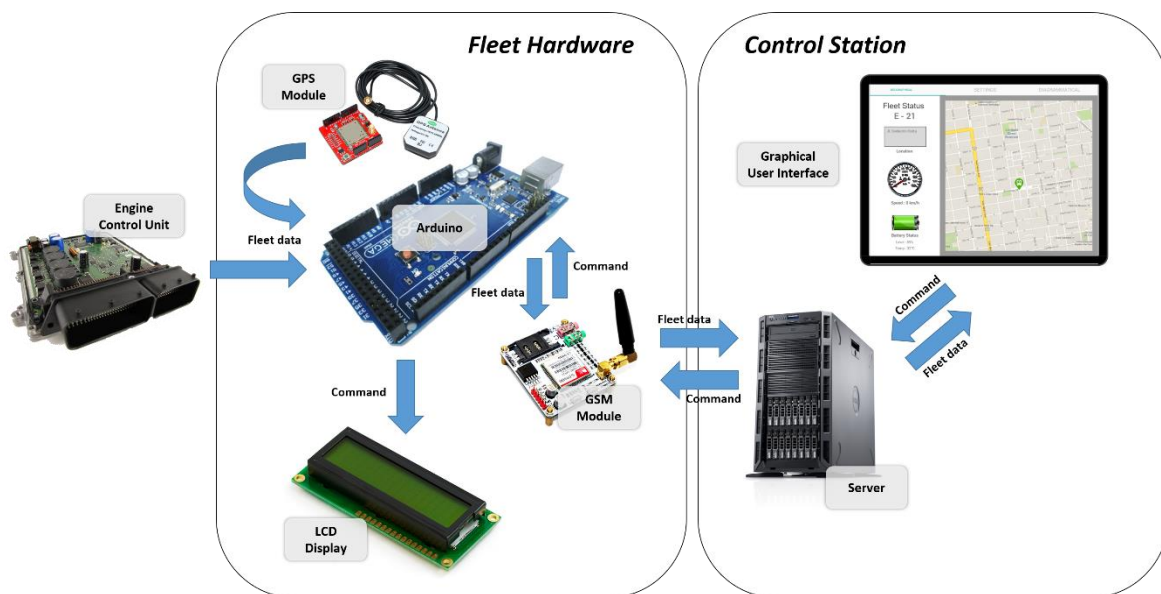
Untuk mengemas seluruh modul dalam *hardware*, dibuat sebuah *packaging* dengan desain seperti berikut.



**Gambar 24 Desain *packaging***

Pada desain tersebut, dibuat lubang-lubang untuk memasukkan modul-modul, kontroller dan *display*. Seluruh modul yang digunakan berada di dalam kotak, sedangkan untuk sumber daya listrik berasal dari luar.

#### **4.5 DESAIN SISTEM KESELURUHAN**



**Gambar 25 Desain keseluruhan sistem ECU *monitoring***

Gambar di atas menunjukkan gambaran desain secara keseluruhan. Sistem ini merupakan gabungan dari desain sub-sistem yang telah dipaparkan sebelumnya. Desain sistem ini juga mengikuti arsitektur yang telah dipilih sebelumnya.

Secara garis besar sistem terbagi menjadi 2 bagian, yaitu ECU *monitoring* dan *control station*. ECU *monitoring* terhubung dengan Engine Control Unit (ECU) *guided bus*. ECU akan mengirimkan data berupa kecepatan dan level baterai pada ECU *monitoring*.



Arduino merupakan otak atau pengendali utama dari ECU *monitoring*. Arduino akan menerima masukan berupa data fisik kendaraan dari ECU serta data posisi kendaraan dari modul GPS. Arduino kemudian akan mengirimkan data yang dimiliki ke *server* dengan menggunakan modul GSM.

Dari server, data kendaraan akan disampaikan ke control station, kemudian diolah di algoritma utama pada Graphical User Interface (GUI). Dalam GUI terdapat 3 algoritma utama, yaitu algoritma utama, algoritma penjadwalan, serta *display data* atau *interface*. Data yang telah diolah kemudian ditampilkan pada *interface* agar bisa dipantau oleh operator.

Apabila tidak terjadi kondisi darurat, perintah untuk maju dan berhenti diinstruksikan oleh algoritma penjadwalan, yang disampaikan kepada *driver* masing-masing armada lewat LCD Display yang terdapat di hardware. Sementara apabila terjadi situasi darurat, operator dapat memberi komando secara manual lewat *interface* dan instruksi dari algoritma penjadwalan dihentikan sementara.

#### **4.6 DESAIN SIMULASI ALGORITMA PENJADWALAN**

Untuk melakukan simulasi fungsional dari sistem ini, dirancang sebuah simulasi. Simulasi ini merupakan simulasi algoritma penjadwalan dan fungsionalitas dari *graphical user interface* yang sudah dibuat. Karena simulasi sistem dengan menggunakan ECU *monitoring* sungguh memerlukan banyak biaya produksi *hardware* dan membutuhkan banyak armada, sehingga simulasi ini merupakan solusi yang tepat untuk mensimulasikan kinerja sistem yang telah dibuat. Simulasi ini akan mensimulasikan kinerja sepuluh ECU *monitoring*.

Pada bagian ini akan dijelaskan mengenai modul-modul yang digunakan untuk simulasi dan metode yang digunakan.

##### **4.6.1 Aplikasi Dummy Data**

Modul pertama adalah aplikasi untuk men-generate *dummy data*. Data yang dihasilkan dari aplikasi ini adalah *fleet data* yang pada implementasinya akan dihasilkan oleh ECU *monitoring*, yaitu data baterai armada (suhu dan energi) serta posisi armada (longitude dan latitude). Aplikasi ini berbasis windows. Aplikasi ini akan menjalankan tugas dari ECU *monitoring*, yaitu mengirimkan *fleet data* ke server. Selain mengirimkan *fleet data*, aplikasi ini juga dapat menerima perintah dari server untuk mensimulasikan armada yang berhenti atau sedang berjalan.

Metode yang digunakan dalam mensimulasikan ECU *monitoring* ini adalah dengan membagi jalur Soekarno-Hatta, Bandung setiap tiga meter per segmen. Setiap bagian ini terdiri dari data *longitude* dan *latitude* yang bersesuaian dengan lokasinya masing-masing. Ada sepuluh armada yang akan disimulasikan, masing-masing armada dimulai di lokasi yang berbeda-beda. Jika aplikasi ini menerima perintah untuk armada berjalan, setiap detiknya, lokasi armada yang bersangkutan akan maju satu segmen. Jika aplikasi ini menerima perintah untuk armada berhenti, lokasi armada yang bersangkutan akan tetap di segmen tersebut. Selain itu, setiap detiknya aplikasi ini akan mengirimkan *fleet data* ke server.

##### **4.6.2 Server**

Pada simulasi ini digunakan server MQTT broker HiveMQ yang tersedia gratis di internet. Server ini berfungsi sebagai pusat yang mengirim dan menerima data dari aplikasi *dummy data* dan dari GUI.



#### **4.6.3 Graphical User Interface (GUI)**

Modul terakhir yang digunakan adalah GUI. GUI yang digunakan dalam *hardware in-the-loop simulation* ini sama dengan GUI pada implementasi.