UNIVERSITAS INDONESIA

RANCANG BANGUN SISTEM PEMANTAUAN CUACA BERGERAK BERBASIS *GOOGLE EARTH* DAN *GOOGLE MAPS*

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

AHMAD FAUZI

040503006Y

FAKULTAS TEKNIK

DEPARTEMEN ELEKTRO

DEPOK

DESEMBER 2009

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri,

dan semua sumber baik yang dikutip maupun dirujuk

telah saya nyatakan dengan benar.

Nama : Ahmad Fauzi

NPM : 040503006Y

Tanda Tangan : 

Tanggal : 28 Desember 2009

LEMBAR PENGESAHAN

Skripsi ini diajukan oleh	:	
Nama	:	Ahmad Fauzi
NPM	:	040503006Y
Program Studi	:	Teknik Elektro
Judul Skripsi	:	Rancang Bangun Sistem Pemantauan Cuaca Bergerak Berbasis *Google Earth* dan *Google Maps*

Telah berhasil dipertahankan dihadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Strata 1 pada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia.

DEWAN PENGUJI

Pembimbing	: Prof. Dr. Ir. Dadang Gunawan, M.Eng	( )
Penguji	: Dr. Ir. Arman Djohan Diponegoro, M.Eng	( )
Penguji	: Filbert Hilman Juwono, S.T., M.T.	( )

Ditetapkan di	:	Depok
Tanggal	:	28 Desember 2009

KATA PENGANTAR

Puji syukur penulis sampaikan kepada Allah SWT atas segala Karunia dan Rahmat-Nya sehingga skripsi ini dapat terselesaikan. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Teknik Departemen Teknik Elektro pada Fakultas Teknik Universitas Indonesia. Saya menyadari bahwa tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai pada penyusunan skripsi ini, sangatlah sulit bagi saya untuk memperoleh gelar sarjana. Oleh karena itu, saya mengucapkan terima kasih kepada :

(1) Prof. Dr. Ir. Dadang Gunawan, M.Eng, selaku dosen pembimbing yang telah menyediakan waktu, tenaga dan pikiran untuk mengarahkan saya dalam penmyusunan skripsi ini.

(2) Orang tua (Tarmidi dan Kusniati) dan adik (Muhamad Wahyudin) yang telah memberikan dukungan moril dan material,

(3) Merry Wahyuningsih yang senantiasa memberikan motivasi dan masukan, dan

(4) Tomy Abu Zairi, Zulfikar Widyaseno, Adytiawan Arga Dwitama, D. Ari Wicaksono, Daniel Ortega, Harry Nofrianz Prakasa, Khotman Hilmy Fajrian, Rinda Airin, Taufiq Alif Kurniawan, dan teman-teman forum Kaskus yang telah memberi berbagai masukan

Akhir kata, semoga Tuhan Yang Maha Esa membalas segala kebaikan semua pihak yang telah membantu penyusunan skripsi ini dengan balasan yang lebih baik. Semoga skripsi ini membawa manfaat yang besar bagi pengembangan ilmu pengetahuan dan teknologi.

Depok, 28 Desember 2009

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI

TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Indonesia, saya yang bertanda tangan dibawah ini:

Nama : Ahmad Fauzi

NPM : 040503006Y

Program Studi : Teknik Elektro

Departemen : Teknik Elektro

Fakultas : Teknik

Jenis Karya : Skripsi

demi perkembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul :

RANCANG BANGUN SISTEM PEMANTAUAN CUACA BERGERAK BERBASIS *GOOGLE EARTH* DAN *GOOGLE MAPS*

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/format-kan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok

Pada tanggal : 28 Desember 2009

Yang menyatakan

(Ahmad Fauzi)

Nama : Ahmad Fauzi

Program Studi : Teknik Elektro

Judul : Rancang Bangun Sistem Pemantauan Cuaca Bergerak berbasis *Google Earth* dan *Google Maps*

Skripsi ini membahas dan menganalisa sistem pemantauan cuaca yang terintegrasi dengan menggunakan *Google Earth* dan *Google Maps*. Sistem ini terdiri atas GPS, sensor suhu dan kelembaban(SHT11), aplikasi *desktop* sebagai pengolah informasi, dan aplikasi *web*site yang terpadu. Aplikasi website berfungsi untuk menampilkan data sensor ke *Google Maps* dan menghasilkan *file* KML untuk *Google Earth*. Hasil dari sistem ini menunjukan penggunaan *Google Earth* lebih interaktif karena tidak membutuhkan koneksi internet yang kontinyu untuk menampilkan peta dan juga kemudahan untuk pemantauan cuaca secara *real time*.

Kata kunci:

GPS, *Google Earth*, Mikrokontroler, KML

ABSTRACT

Name : Ahmad Fauzi

Study Program : Teknik Elektro

Title : Mobile Weather Monitoring System Design Based on Google Earth and Google Maps

This thesis discusses and analyzes the weather monitoring system that integrates with Google Earth and Google Maps. This system consists of GPS, temperature and humidity sensors (SHT11), desktop application to process information, and integrated web application. Web application used to display sensor data for Google Maps and generated the KML file for Google Earth. The system's result show the use of Google Earth is more interactive because it do not require continuous internet connection to display maps and easy to use for monitoring application in real time.

Key words:

GPS, Google Earth, Microcontroller, KML

DAFTAR ISI

HALAMAN SAMPUL i
HALAMAN PERNYATAAN ORISINALITAS ii
LEMBAR PENGESAHAN iii
KATA PENGANTAR iv
HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI v
TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS v
ABSTRAK vi
ABSTRACT vii
DAFTAR ISI viii
DAFTAR GAMBAR x
DAFTAR TABEL xii
PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Tujuan 3
1.3 Manfaat 3
1.4 Batasan Masalah 3
1.5 Sistematika Penulisan 4
INTEGRASI DENGAN SISTEM PEMANTAUAN LINGKUNGAN DENGAN SISTEM INFORMASI GEOGRAFI (SIG) 6
2.1 Sistem Informasi Geografi 6
2.2 Komponen Penyusun Sistem Pemantauan Cuaca 7
2.2.1. Komponen Data 7
2.2.2. Komponen Penyimpanan dan Pemanggilan Data 10
2.2.3. Komponen Analisis dan Manipulasi Data 11
2.2.4. Komponen Penyajian Data 14
PERANCANGAN SISTEM PEMANTAUAN CUACA 19
3.1 Bagian Sensor 19
3.1.1. Sensor SHT11 19
3.1.2. Bagian Sensor Posisi (GPS) 25
3.2 Lingkungan Pemrograman Delphi 25
3.3. Aplikasi Website, KML, dan Google Earth 28
3.4 Sistem Keseluruhan 30
3.4.1 Sistem Pemantau 30
3.4.2 *Server* 31
3.4.3 *Client* 31
ANALISIS KINERJA SISTEM PEMANTAUAN CUACA 32
4.1 Prototype Sistem Pemantauan Cuaca 32
4.2 Uji Waktu Muat (Loading Time) 36

4.2.1. Uji Muat Waktu Muat Aplikasi Google Earth 36
4.2.2. Uji Muat Waktu Aplikasi Google Maps 39
4.3 Uji Perbandingan 43
4.4 Uji Lapangan 45
4.5 Aplikasi Multi Sensor 47
KESIMPULAN 50
DAFTAR REFERENSI 51
DAFTAR PUSTAKA 52
DAFTAR LAMPIRAN 53

DAFTAR GAMBAR

Gambar 1.1. Sistem yang diajukan untuk pemantauan cuaca……2

Gambar 2.1. Dimensi Sensor SHT11……..…….... 8

Gambar 2.2 Komponen Penyusun data pada sensor ……...……..10

Gambar 2.3 MySQL yang terintegrasi dengan PHPMyAdmin…… 11

Gambar 2.4 Delphi dengan Komponen TGPS……...12

Gambar 2.5 Board DTR-AVR……...13

Gambar 2.6 Piranti Lunak *Google Earth*……...15

Gambar 2.7. Mekanisme Network link……..17

Gambar 3. 1 Konfigurasi ATMEGA 8535 dengan SHT11……20

Gambar 3. 2 Skema Sistem Minimum DTR-AVR ……...21

Gambar 3. 3 Diagram Alir Program Pada Mikrokontroler……23

Gambar 3. 4 Diagram Alir untuk Membaca Suhu……24

Gambar 3. 5 Garmin PC 18 USB……...25

Gambar 3. 6 Piranti Lunak pada Komputer Kontrol……..26

Gambar 3. 7 Diagram Alir pada Program Pengolah di Komputer Kontrol……...27

Gambar 3. 8 Sistem Pengolahan Database……28

Gambar 3. 9 Diagram Alir pada Aplikasi *Google Maps* dan *Google Earth*……..29

Gambar 3. 10 Rancangan Sistem Pemantauan……...30

Gambar 4. 1 Prototype (Purwarupa) Sistem Hasil Rancangan……32

Gambar 4. 2 Antar muka Aplikasi GPS pada Komputer……...33

Gambar 4. 3 Antarmuka Aplikasi Sensor SHT11 pada Komputer……...34

Gambar 4. 4 Aplikasi *Google Maps*……35

Gambar 4. 5 Aplikasi *Google Earth*……..36

Gambar 4. 6 Grafik Uji Muat Aplikasi *Google Earth*……...37

Gambar 4. 7 Grafik TCP IP IO Graph pada Wireshark……38

Gambar 4. 8 Grafik Throughput Aplikasi *Google Earth*……..39
Gambar 4. 9 Percobaan Waktu Muat (Loading Time) *Google Maps*……40
Gambar 4. 10 Grafik Rata-Rata Waktu Muat Setiap Percobaan……40
Gambar 4. 11 Grafik TCP IO Graph dari *Google Maps*……..…..41
Gambar 4. 12 Grafik Throughput pada *Google Maps* ……..….42
Gambar 4.13 Grafik Perbandingan Suhu SHT11, Suhu AWS,
dan Termometer Bola Basah-Bola Kering ……....43
Gambar 4.14 Grafik Perbandingan Kelembaban SHT11, Suhu AWS,
dan Termometer Bola Basah-Bola Kering…….....44
Gambar 4. 15 Simulasi Percobaan aplikasi Multi sensor……..….47
Gambar 4. 16 Percobaan pada aplikasi multiclient yang dilihat pada sisi *client*...48

DAFTAR TABEL

Tabel 3. 1 Tabel Koneksi DT-AVR dengan Modul SHT11……..22

Tabel 4. 1 Validasi per lokasi di Kampus UI Depok terhadap *Google Earth*……45

Tabel 4.2 Perbanding Kinerja Pemantauan Cuaca Melalui *Google Earth* dan *Google Maps*……..…….49

BAB 1

PENDAHULUAN

1.1 Latar Belakang

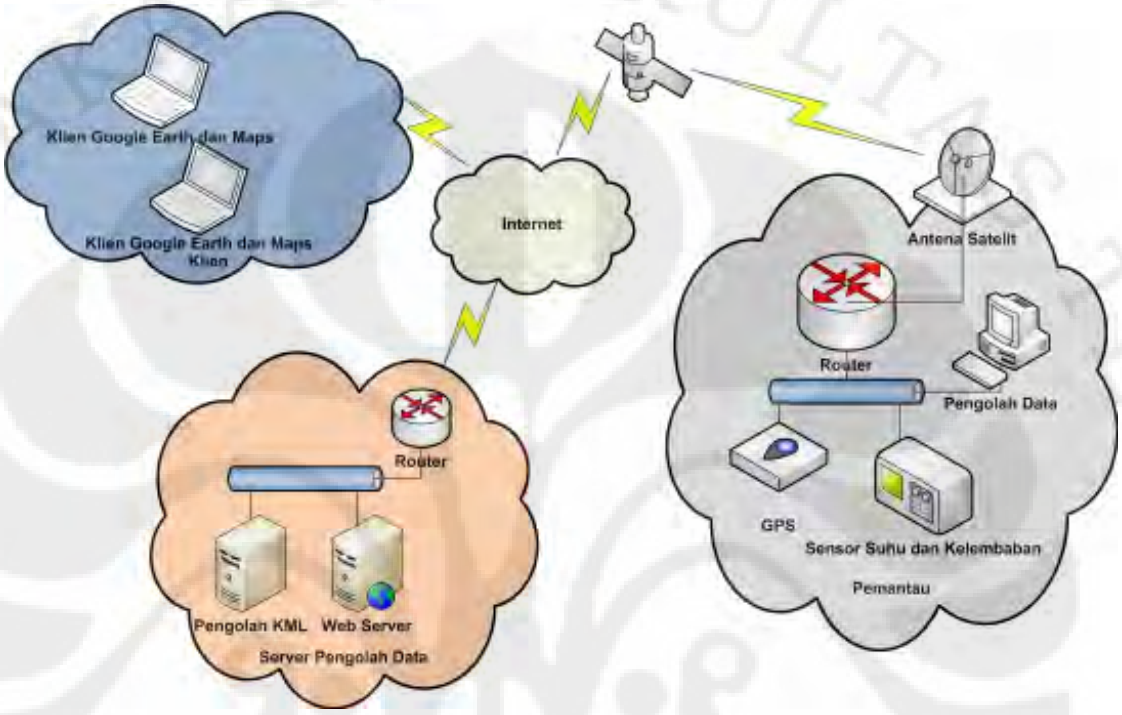
Sistem Informasi Geografi sekarang ini merupakan salah satu kebutuhan dalam kehidupan sehari-hari. Aplikasi sistem informasi geografi banyak dan beragam, mulai dari untuk penanganan bencana, survey geografi, sampai dengan sekedar untuk mengetahui letak rumah teman kita sendiri. Sistem informasi geografi mengintegrasikan antara perangkat keras, perangkat lunak, dan data untuk ditangkap, diolah, dianalisis, dan ditampilkan dalam berbagai bentuk informasi geografi [1].

Sistem Informasi Geografi memudahkan untuk memahami, intepretasi, dan menyajikan data yang mempunyai hubungan dengan peta, globe, laporan, dan peta. Sehingga dapat dipahami bahwa sistem informasi geografi merupakan kombinasi berbagai teknologi untuk menyajikan suatu dalam sebuah visualisasi tentang keadaan permukaan bumi.

Perkembangan Sistem Informasi Geografi (SIG) didukung oleh berkembangnya berbagai aplikasi yang mendukung Sistem Informasi Geografi (SIG) seperti *Arc View* dan *Google* serta perkembangan internet yang mulai menuju jaringan pita lebar (*broadband*). Visualisasi dalam penyajian data merupakan hal kebutuhan dalam sistem ini untuk mendapatkan data spasial yang terletak berdasarkan objek pantauan berada. Penggunaaan piranti lunak seperti *Google Earth* mampu mendukung visualiasi dan juga pembaharuan (*update*) dari objek yang dipantau sehingga aplikasi SIG dapat interaktif.

Penggunaan teknologi informasi dan komunikasi dapat digunakan untuk pemantuan cuaca secara *real time.* Salah satu aplikasi Sistem Informasi Geografi (SIG) yang banyak digunakan untuk pemantauan adalah *Google Earth*. Fitur-fitur yang ada di dalamnya dapat digunakan untuk aplikasi pemantauan cuaca dengan modifikasi data yang ada dalam KLM (*Keyhole Markup Language*). Diharapkan

dengan integrasi ini pengguna dapat mendapatkan informasi mengenai lingkungan secara cepat dan interaktif.



Gambar 1.1. Sistem yang diajukan untuk pemantauan cuaca

Sistem yang diajukan oleh penulis dapat dilihat pada gambar 1.2. Sistem ini terdiri atas 4 bagian yang dapat dijelaskan sebagai berikut :

1. Sistem pada cuaca menggunakan GPS dan sensor kelembaban serta suhu. Posisi dari cuaca diperoleh dari konstelasi satelit NAVSTAR GPS secara periodik.
2. Setelah data diterima, aplikasi piranti lunak yang ada pada cuaca mengolah data kemudian di kirimkan melalui internet satelit komunikasi
3. Satelit Komunikasi kemudian melanjutkan ke jaringan internet dan data dituliskan ke dalam database *server* untuk di simpan dan diolah lebih lajut
4. *Server* penerima kemudian dapat diakses melalui *website* yang terdapat *Google Maps* dan aplikasi *Google Earth.*

1.2 Tujuan

Tujuan dari skripsi ini adalah :

1. Rancang bangun sistem pemantauan cuaca yang meliputi suhu, kelembaban udara, *dewpoint*, dan kondisi udara (berkabut/tidak berkabut) yang terintegrasi dengan sistem informasi Geografi (SIG) dengan memanfaatkan *Google Earth* dan *Google Maps*
2. Perbandingan performa *Google Earth* dan *Google Maps* pada sistem pemantauan lokasi
3. Memvalidasi sensor SHT11 untuk dapat digunakan di lapangan
4. Menganalisis kemampuan sistem ini untuk dapat digunakan di lapangan dengan percobaan di Kampus UI Depok untuk aplikasi *Google Earth* dan *Google Maps*.

1.3 Manfaat

Manfaat dari skripsi ini adalah :

1. Sebagai purwarupa (*prototype*) yang nantinya bisa gunakan sebagai salah satu acuan dalam pemantauan cuaca
2. Sebagai acuan pemanfaatan peta *Google Earth* dan *Google Maps* dalam pemantauan cuaca
3. Dapat dikembangkan sebagai *prototype* untuk sistem pemantauan lainnya seperti untuk tsunami, kebakaran hutan, dan banjir

1.4 Batasan Masalah

Batasan skripsi ini adalah membuat sistem pemantauan cuaca yang terintegrasi secara *real time* dengan memanfaatkan sensor suhu dan kelembaban udara (SHT11), GPS, dan penggunaan KLM dalam *Google Earth* dan *Google Maps*. Pemantauan pada *Google Earth* mencakup posisi dan parameter lingkungan (suhu dan kelembaban) sedangkan *Google Maps* dibatasi hanya pada posisi. Bahasa Pemrograman yang dipakai adalah bahasa C untuk mikrokontroler dan Delphi untuk Aplikasi *Desktop*. Sedangkan pangkalan data (*database*)

menggunakan MySQL dan aplikasi *Website* menggunakan kombinasi dari PHP, XML, AJAX dan HTML.

1.5 Sistematika Penulisan

Penulisan dalam skripsi ini terdiri dari 5 bab. Ringkasan dari tiap Bab adalah sebagai berikut

BAB 1 PENDAHULUAN

Bab 1 membahas tentang latar belakang dari pengembangan skripsi ini, tujuan yang akan diperoleh dan manfaatnya serta pembatasan masalah dari skripsi yang dibuat

BAB 2 INTEGRASI SISTEM PEMANTAUAN CUACA DENGAN SISTEM INFORMASI GEOGRAFI (SIG)

Bab 2 ini membahas landasan teori yang menjadi rujukan dari skripsi ini. Pembahasan garis besarnya adalah tentang sistem informasi geografi tetapi lebih terfokus bagaimana sistem di dalam berinteraksi antara piranti lunak dan keras yang ada untuk sistem pemantauan lingkungan. Komponen-komponen dalam sistem informasi yang dibahas lebih ditekankan pada sisi teknis seperti pembuatan piranti lunak dan piranti keras yang digunakan dalam aplikasi ini.

BAB 3 PERANCANGAN SISTEM

Bab 3 ini membahas rancangan dari sistem ini. Bab ini memuat bagaimana rancangan interaksi antara piranti lunak dan piranti keras dari sistem yang akan dibangun disertai dengan bagan dan gambar yang mendukung penjelasannya.

BAB 4 UJI KINERJA DAN ANALISIS

Bab 4 merupakan uji kinerja dan analisis dari sistem yang telah dibuat. Uji kerjanya antara lain uji per bagian dari sistem dan uji keseluruhan dari sistem. Uji per bagian dari sistem untuk menguji setiap sub sistem apakah berfungsi dengan baik. Uji keseluruhan dari sistem menguji kinerja dari sistem untuk berfungsi secara keseluruhan. Uji ini meliputi waktu muat sistem, kinerja pembaharuan data,

dan perbandingan antara sensor digital dan sensor analog suhu dan kelembaban udara.

BAB 5 KESIMPULAN

Bab 5 berisi kesimpulan dari skripsi yang di buat yang disarikan dari uji kinerja dan analisis serta konsep keseluruhan dari sistem informasi geografis (SIG) ini. Kesimpulan juga berisi rekomendasi sistem yang dapat dikembangkan selanjutnya.

BAB 2

INTEGRASI DENGAN SISTEM PEMANTAUAN CUACA DENGAN SISTEM INFORMASI GEOGRAFI (SIG)

2.1 Sistem Informasi Geografi

Sistem Informasi Geografi (SIG) atau *Geographic Information System* (GIS) adalah suatu sistem informasi yang dirancang untuk bekerja dengan data yang bereferensi spasial atau berkoordinat geografi atau dengan kata lain suatu SIG adalah suatu sistem basis data dengan kemampuan khusus untuk menangani data yang bereferensi keruangan (spasial) bersamaan dengan seperangkat operasi kerja [6]. Tujuan pokok dari pemanfaatan Sistem Informasi Geografis adalah untuk mempermudah mendapatkan informasi yang telah diolah dan tersimpan sebagai atribut suatu lokasi atau obyek. Ciri utama data yang bisa dimanfaatkan dalam Sistem Informasi Geografis adalah data yang telah terikat dengan lokasi dan merupakan data dasar yang belum dispesifikasi [7].

Lukman [8] menyatakan bahwa sistem informasi geografi menyajikan informasi keruangan beserta atributnya yang terdiri dari beberapa komponen utama yaitu:

1. Masukan data merupakan proses pemasukan data pada komputer dari peta (peta topografi dan peta tematik), data statistik, data hasil analisis penginderaan jauh data hasil pengolahan citra digital penginderaan jauh, dan lain-lain. Data-data spasial dan atribut baik dalam bentuk analog maupun data digital tersebut dikonversikan kedalam format yang diminta oleh perangkat lunak sehingga terbentuk basisdata (*database*). Basis data adalah pengorganisasian data yang tidak berlebihan dalam komputer sehingga dapat dilakukan pengembangan, pembaharuan, pemanggilan, dan dapat digunakan secara bersama oleh pengguna.
2. Penyimpanan data dan pemanggilan kembali *(data storage* dan *retrieval)* ialah penyimpanan data pada komputer dan pemanggilan kembali dengan

cepat (penampilan pada layar monitor dan dapat ditampilkan/cetak pada kertas).

3. Manipulasi data dan analisis ialah kegiatan yang dapat dilakukan berbagai macam perintah misalnya *overlay* antara dua tema peta, membuat *buffer zone* jarak tertentu dari suatu area atau titik dan sebagainya. Manipulasi dan analisis data merupakan ciri utama dari SIG. Kemampuan SIG dalam melakukan analisis gabungan dari data spasial dan data atribut akan menghasilkan informasi yang berguna untuk berbagai aplikasi
4. Pelaporan data adalah dapat menyajikan data dasar, data hasil pengolahan data dari model menjadi bentuk peta atau data tabular. Menurut Barus dan Wiradisastra [9] Bentuk produk suatu SIG dapat bervariasi baik dalam hal kualitas, keakuratan dan kemudahan pemakainya. Hasil ini dapat dibuat dalam bentuk peta-peta, tabel angka-angka, teks di atas kertas atau media lain (*hard copy*), atau dalam cetak lunak (seperti *file* elektronik).

2.2 Komponen Penyusun Sistem Pemantauan Cuaca

Berdasarkan definisi diatas maka dapat diketahui komponen-komponen yang dapat digunakan dalam penyusunan sistem informasi geografis untuk pemantauan cuaca dan parameter cuaca laut

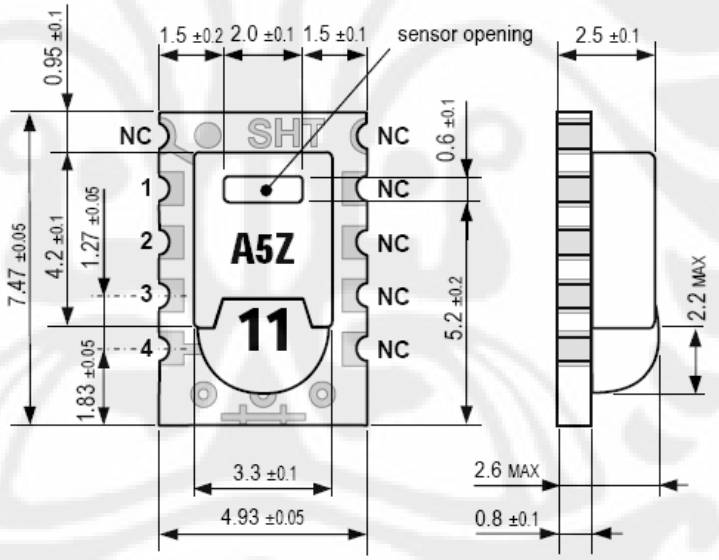
2.2.1 Komponen Data

Data merupakan hal yang akan diolah dalam sistem informasi geografi. Data dapat berupa kondisi permukaan bumi dan gejala yang ada. Komponen data yang diambil dapat menggunakan sensor analog maupun digital. Sensor analog mempunyai kecenderungan presisi yang lebih tepat namun susah untuk diinterpretasikan secara digital karena harus dikonversikan terlebih dahulu ke dalam data digital untuk pembacaan yang lebih teliti. Kemudahan dalam penginterpretasian data dalam bentuk digital merupakan merupakan salah satu keunggulan dari sensor digital. Konsep konfigurasi sensor yang digunakan mengacu pada sensor *web*[10]. Sensor *web* sendiri adalah jenis jaringan sensor atau sistem informasi geografis yang cocok digunakan untuk pemantauan

cuaca[11]. Sensor *web* terdiri atas sejumlah *pods* dengan masing-masing *pods* terdiri atas[12]:

1. Satu atau lebih sensor atau kanal data
2. Unit pemroses data seperti mikrokontroler atau mikroprosesor
3. Komunikasi dua arah seperti seperti radio dan antena
4. Sumber energi seperti baterai yang dikombinasikan dengan sel surya
5. Bahan pelindung untuk melindungi sensor terhadap cuaca yang ekstrim

Salah satu sensor yang digunakan dalam pemantauan cuaca adalah SHT11 yang merupakan sensor suhu dan kelembaban. Sensor ini telah dikalibrasi oleh perusahaan pembuatnya, Sensirion Inc. Dimensi dari sensor ini dapat dilihat pada gambar 2.1.



Gambar 2.1. Dimensi Sensor SHT11[13]

Komponen data penting lainnya adalah posisi. Sampai sekarang, penentuan posisi yang paling tepat adalah dengan menggunakan GPS. *Global positioning system* (GPS) adalah salah satu sistem navigasi satelit yang berfungsi dengan baik. Sistem ini menggunakan 24 satelit yang mengirimkan sinyal gelombang mikro ke Bumi. Sinyal ini diterima oleh alat penerima di permukaan, dan digunakan untuk menentukan posisi, kecepatan, arah, dan waktu. Sistem yang

serupa dengan GPS antara lain GLONASS Rusia, Galileo Uni Eropa, IRNSS India[14].

Penggunaan GPS antara lain digunakan untuk [15] :

1. Militer

   GPS digunakan untuk keperluan perang, seperti menuntun arah bom, atau mengetahui posisi pasukan berada. Dengan cara ini maka kita bisa mengetahui mana teman mana lawan untuk menghindari salah target, atau menentukan pergerakan pasukan.
2. Navigasi

   GPS banyak juga digunakan sebagai alat navigasi seperti kompas. Beberapa jenis kendaraan telah dilengkapi dengan GPS untuk alat bantu navigasi, dengan menambahkan peta, maka bisa digunakan untuk memandu pengendara, sehingga pengendara bisa mengetahui jalur mana yang sebaiknya dipilih untuk mencapai tujuan yang diinginkan.
3. Sistem Informasi Geografis

   Untuk keperluan Sistem Informasi Geografis, GPS sering juga diikutsertakan dalam pembuatan peta, seperti mengukur jarak perbatasan, ataupun sebagai referensi pengukuran.
4. Sistem pelacakan kendaraan

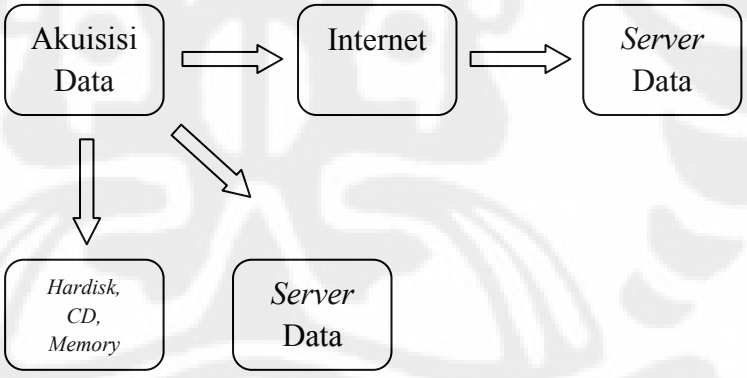
   Kegunaan lain GPS adalah sebagai pelacak kendaraan, dengan bantuan GPS pemilik kendaraan/pengelola armada bisa mengetahui ada dimana saja kendaraannya/aset bergeraknya berada saat ini.
5. Pemantau gempa

   GPS dengan ketelitian tinggi bisa digunakan untuk memantau pergerakan tanah, yang ordenya hanya mm dalam setahun. Pemantauan pergerakan tanah berguna untuk memperkirakan terjadinya gempa, baik pergerakan vulkanik ataupun tektonik

2.2.2. Komponen Penyimpanan dan Pemanggilan Data

Sistem penyimpanan dalam sistem informasi geografi telah berkembang dengan cepat. Mulai dari penyimpan data konvensional seperti kertas sampai penyimpanan dalam bentuk data digital misalnya dalam bentuk cakram padat (CD), *Flash Disk, Hardisk*, dan Kartu Data (*MMC Card*). Komponen data bisa diolah dalam suatu sistem penyimpanan melalui suatu koneksi tertentu misalnya koneksi langsung atau tidak langsung misalnya melalui internet dengan menyimpan data melalui *server* tertentu.

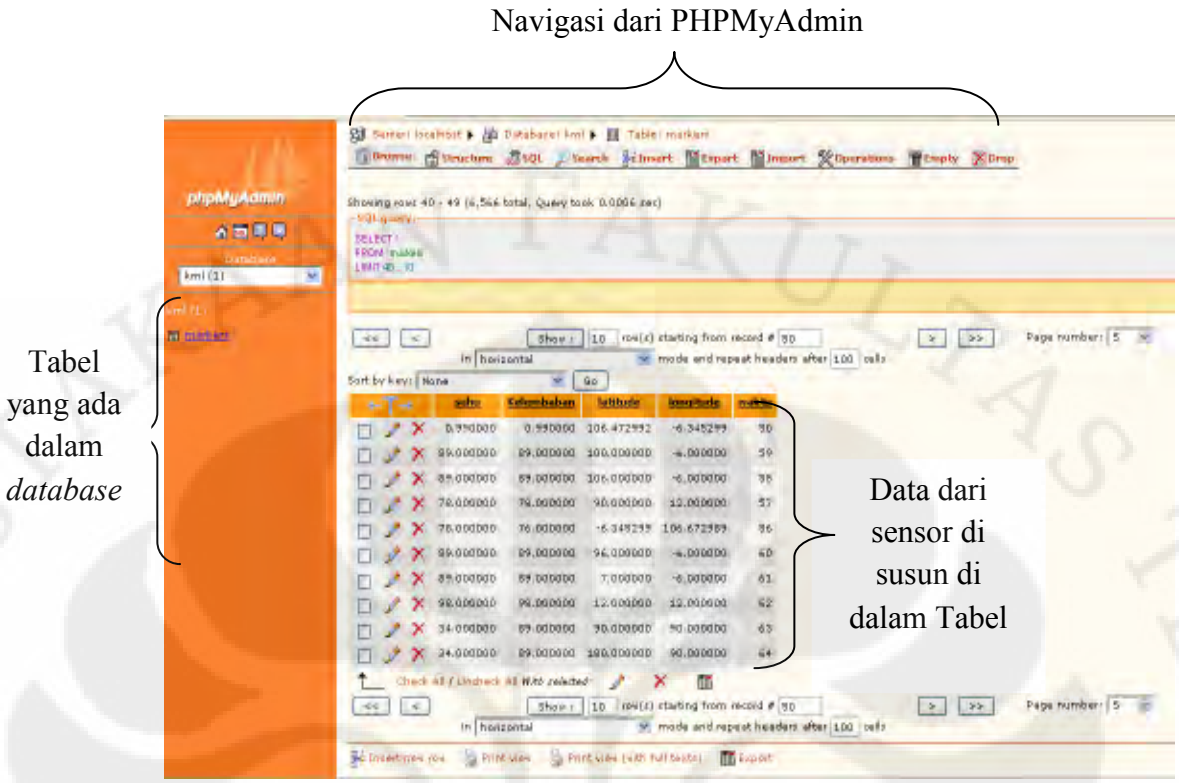
Setelah sistem tersimpan dalam suatu *server*. Data dapat diolah kembali dalam suatu pangkalan data (*database*) yang lebih teratur misalnya dengan menggunakan *database server*. Database *server* yang cukup terkenal dan gratis adalah MySQL. Dalam MySQL data dapat diklasifikasikan menurut urutan tertentu sehingga memudahkan dalam pemanggilan kembali data. Sistemnya dapat disederhanakan seperti pada gambar 2.2.



Gambar 2.2 Komponen Penyusun data pada sensor

Sistem akuisi pada gambar 2.2 merupakan sistem terpadu yang mengintegrasikan antara piranti keras pada sensor dan piranti lunak untuk pengolahan data serta *server* mysql untuk penyimpanan data dengan menggunakan koneksi internet.

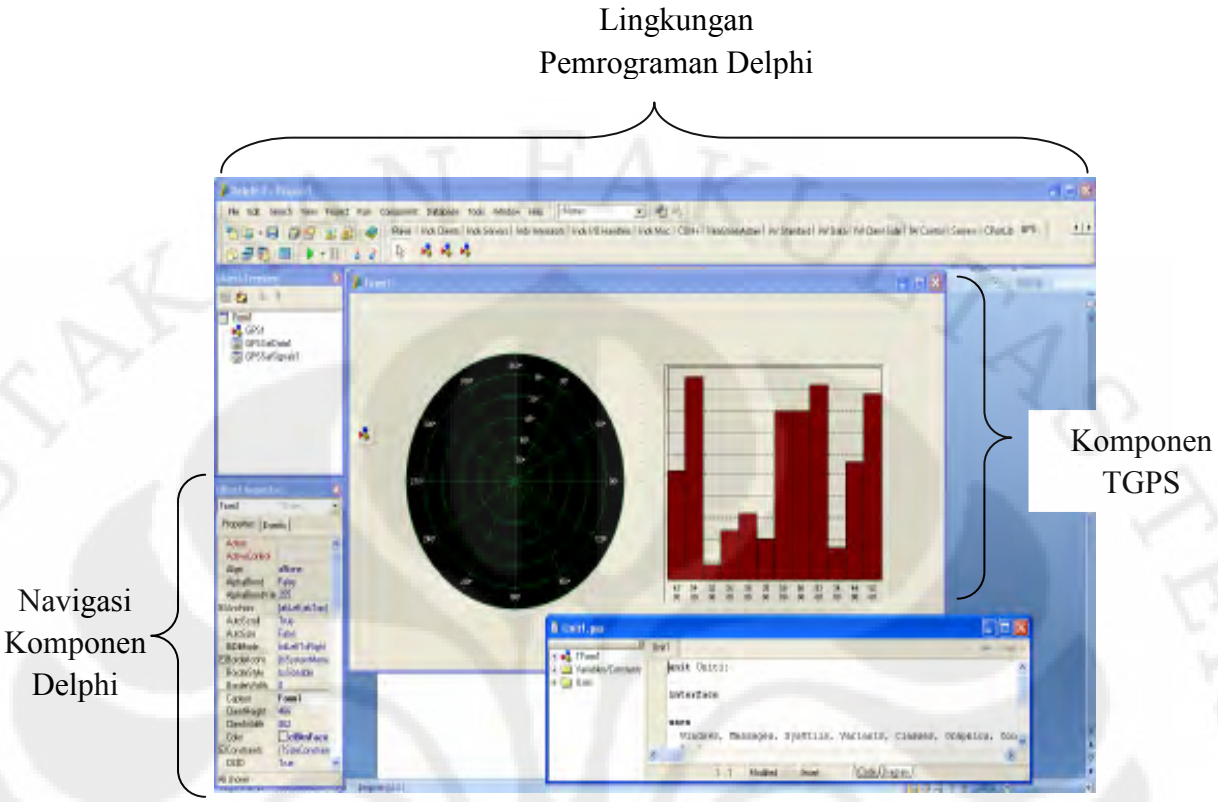
Gambar 2.3 menggambarkan database *server* XAMPP yang terdapat di dalamnya database mysql dan juga terintegrasi dengan PHPMyAdmin



Gambar 2.3 MySQL yang terintegrasi dengan PHPMyAdmin[16]

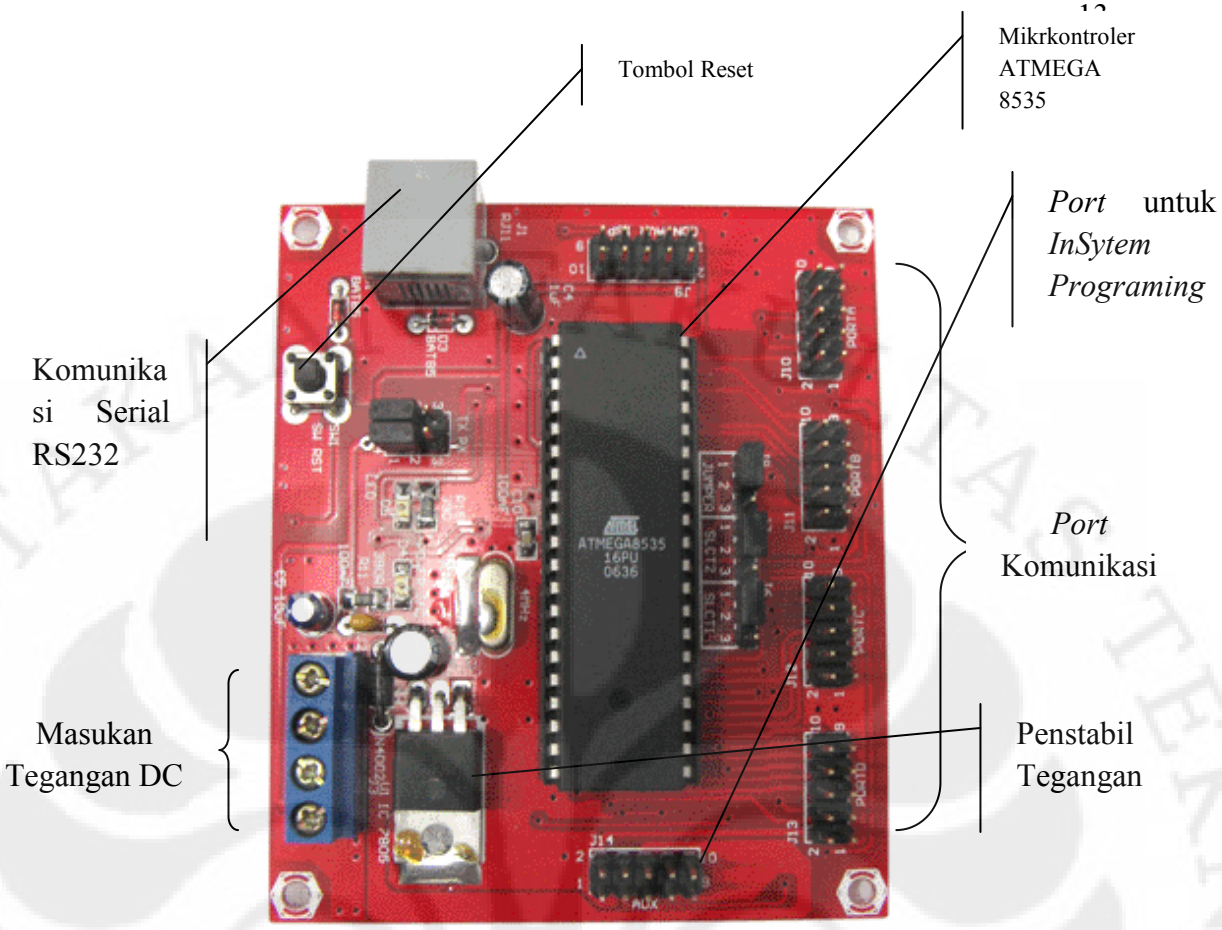
2.2.3 Komponen Analisis dan Manipulasi Data

Komponen analisis dan manipulasi secara sederhana mampu menginterpretasikan hasil dari pemantauan data sehingga dapat digunakan oleh pembaca. Komponen ini berupa piranti lunak yang ada dalam *desktop* untuk menerima hasil dari akuisi data. Data kemudian digabungkan dengan metode tertentu dengan data lain untuk mendapatkan hasil yang diinginkan. Piranti lunak bisa merupakan piranti yang berbayar seperti *Arc View* maupun aplikasi sederhana yang bisa dibuat dengan piranti lunak pemrograman tertentu misalnya Visual *Basic.NET* atau *Delphi*. Komponen-komponen tertentu dari lingkungan pemrogramanDelphi mempunyai kemampuan untuk melakukan pengolahan Sistem Informasi Geografi seperti *Easy Map* . TGPS adalah komponen yang mampu mengakuisi data dari GPS untuk kemudian diolah lebih lanjut.



Gambar 2.4 Delphi dengan Komponen TGPS [17]

Untuk melakukan interpretasi dari sensor SHT11 untuk mendapatkan data suhu yang dapat dipahami dalam satuan Celcius dan Persen maka digunakan mikrokontroler ATMEGA8535 dengan menggunakan sistem minimum DT-AVR *Low Cost Micro System* yang telah dibuat oleh *Innovative Electronic*. Sistem minimum ini telah dilengkapi dengan komunikasi serial RS232 sehingga memudahkan dalam pembangunan sistem informasi geografi ini Lingkungan pemrogramanDelphi seperti terlihat pada gambar 2.4 dipilih karena kemudahan dalam melakukan pemrograman tertanam (*embedded*)



Gambar 2.5 Board DTR-AVR [18]

Komponen sistem minimum pada gambar 2.5 merupakan board dari DT-AVR mempunyai spesifikasi [19]

1. Mendukung varian AVR® 40 pin antara lain: AT90S8535, ATmega8535L, ATmega16(L), ATmega8515(L), AT90S8515, dan ATmega162(L) (Seri AVR® yang tidak memiliki ADC membutuhkan *converter socket*)
2. Memiliki fasilitas *In-System Programming* untuk IC yang mendukung, dilengkapi *LED Programming Indicator*
3. Memiliki hingga 35 pin jalur *input/output*
4. Lengkap dengan osilator 4 MHZ dan memiliki kemampuan komunikasi Serial UART RS-232 yang sudah disempurnakan
5. Lengkap dengan rangkaian reset, tombol manual reset, dan *brown-out detector*

6. Menggunakan tegangan input 9 - 12 VDC dan memiliki tegangan output 5 VDC

2.2.4 Komponen Penyajian Data

Google Earth merupakan salah satu aplikasi peta dan geografi yang sangat populer sekarang ini karena kemampuannya untuk merepresentasikan bentuk muka bumi secara interaktif dan terintegrasi dengan data yang terbaru. Aplikasi *Google Earth* ini dapat digunakan untuk sistem informasi geografis (SIG) suatu wilayah tertentu. Gambar dari aplikasi *Google Earth* dapat dilihat di gambar 2.6.

Tampilan dari *Google Earth* adalah sebagai berikut [20]:

1 Sistem dan Proyeksi Koordinat
   a Sistem koordinat internal *Google Earth* merupakan koordinat geografi dalam bentuk tunggal Sistem Geodetik Dunia tahun 1984 (WGS84).
   b *Google Earth* menampilkan dunia seperti dilihat dari pesawat atau satelit yang mengorbit. Proyeksi ini digunakan untuk memperoleh efek yang disebut Perspektif Umum. Ini mirip dengan proyeksi Ortografi, kecuali titik perspektifnya merupakan jarak terbatas (dekat bumi) daripada jarak tidak terbatas (luar angkasa).
2 Resolusi dasar
   a Amerika Serikat: 15 m (beberapa negara bagian 1 m atau lebih baik)
   b Andorra, Belanda, Britania Raya, Denmark, Jerman, Liechtenstein, Luksemburg,San Marino, Swiss, Vatikan: 1 m atau lebih baik
   c Seluruh dunia: Umumnya 15 m (beberapa area, seperti Antartika, resolusinya sangat rendah), tetapi ini tergantung pada kualitas satelit/fotografi udara yang diunggah.
3 Resolusi tinggi
   a Amerika Serikat: 1 m, 0.6 m, 0.3 m, 0.15 m (sangat jarang, contohnya Cambridge dan *Google* Campus, atau Glendale)

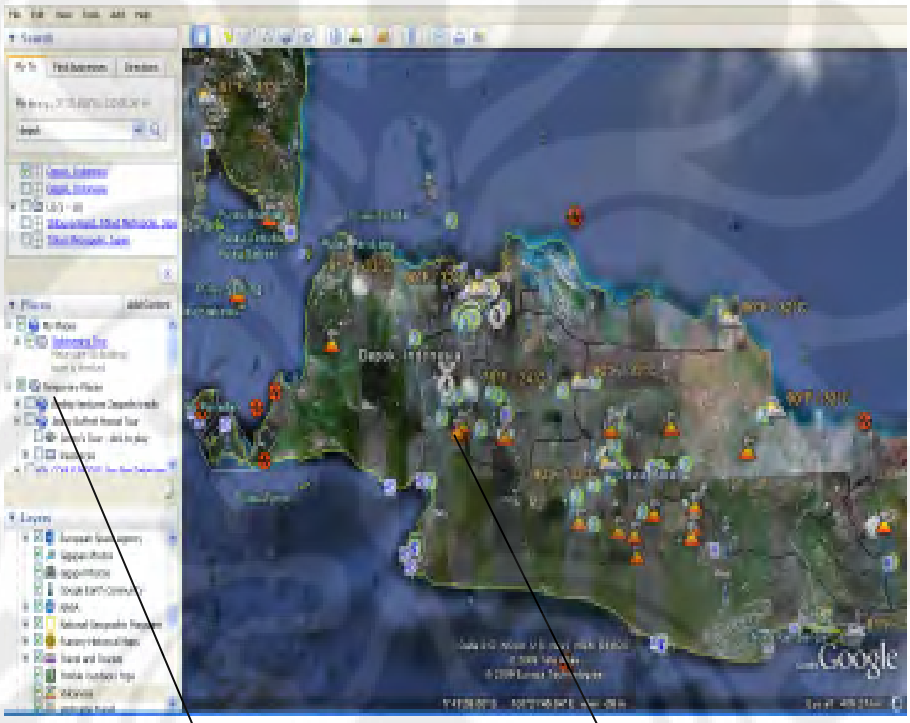
b Eropa: 0.3 m, 0.15 m (contohnya Berlin, Hamburg, Zürich)

4 Resolusi ketinggian

a Permukaan: bervariasi menurut negara

b Dasar laut: Tidak tersedia (sebuah skala warna memperkirakan kedalaman dasar laut "diperlihatkan" pada permukaan).

5 Umur: Gambar kadang-kadang kurang dari 3 tahun. Tanggal selanjutnya untuk informasi hak cipta sering dirujuk sebagai tanggal dimana gambar diambil, tetapi praktik ini tidak benar.



Gambar 2.6 Piranti Lunak *Google Earth* [21]

Pada KML adalah format *file* yang digunakan untuk menampilakan data geografi dalam laman earth seperti *Google Earth*, *Google Maps*, dan *Google Maps* untuk aplikasi *mobile*. *File* KLM diolah dalam cara yang sama dengan HTML dan XML yang diproses oleh daring *web*. Seperti HTML, KML juga berdasarkan struktur tag dengan nama dan atribut untuk tujuan penampilan yang

sesuai dengan tujuannya. *Google Earth* dan *Maps* akan berperan sebagai daring pada *file* KML. *File* KML berekstensi .kml yang merupakan *file* dari *Google Earth* yang mengarahkan lokasi tertentu tergantung posisi yang ada pada *file* kml tersebut.

KML dapat digunakan untuk mendukung aplikasi

1. Menentukan ikon dan label untuk mengidetifikasi lokasi dalam permukaan planet
2. Mengkreasikan beberapa posisi kamera untuk mendefiniskan jangkauan tertentu tergantu keinginan dan fitur yang kita gunakan
3. Meng-overlay gambar ke tanah atau layar
4. Mendefiniskan penampakan tertentu
5. Menulis deskripsi dengan HTML, termasuk tautan dan gambar yang tertanam(*embedded*)
6. Penggunaan folder untuk pengelompokan secara hirarkis fitur-fitur.
7. Secara dinamik mengupdate KML *file* dari *remote* atau lokasi jaringan lokal
8. Menampilkan tekstur COLLADA untuk objek-objek tiga dimensi.

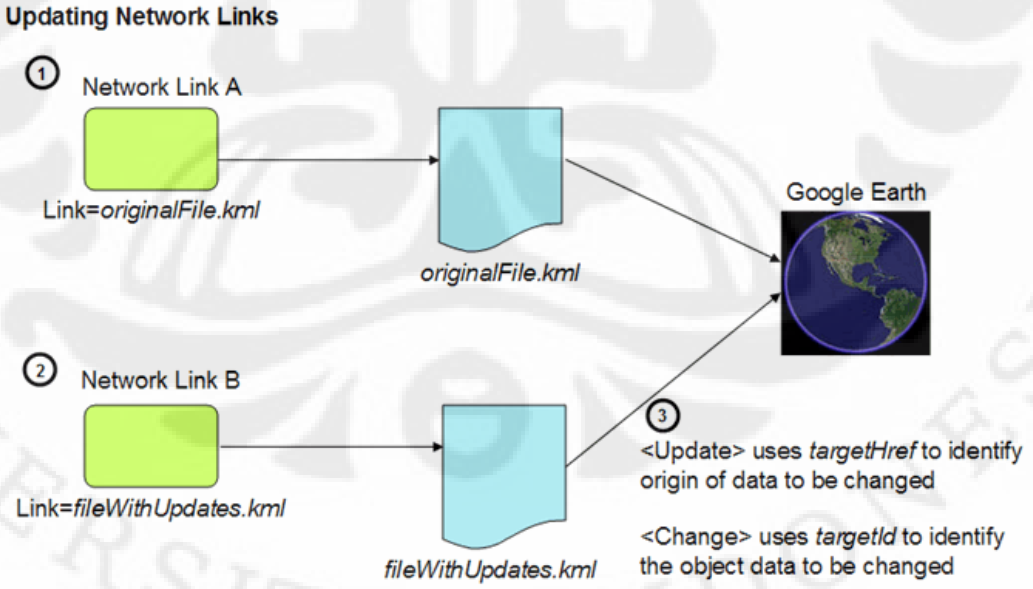
Untuk menampilkan sensor yang telah digunakan maka digunakan *Google Earth* sebagai salah satu aplikasi dari sistem informasi Geografis (SIG) sehingga bisa diperoleh data yang *real time*. Aplikasi ini dapat terjadi karena menggunakan KML yang berdasarkan bahasa dari XML.

Setelah *file* KML telah di*unduh (download)*, maka pengguna tidak harus kembali meng*unduh (download) file* tersebut untuk mendapatkan *update* dari suatu konten yang telah berubah misalnya *real time* data dari sensor dan posisi dari GPS.

Networklink pada KML memungkinkan untuk memperbaharui konten dalam waktu tertentu. Fungsi yang digunakan antara lain **refreshMode** dan **refreshInterval** atau **expires**. Cara kerjanya adalah sebagai berikut [22]:

1. Sebuah *Networklink* me*load file* KLM asli ke dalam *Google Earth*. Sebuah elemen nantinya akan diperbarui membutuhkan identitas eksplisist identitas (id) yang mendefiniskan ketika itu pertama kali di spesifisikan. Identitas-identias tersebut harus unik untuk *file* yang akan didefinsikan..
2. *Networklink* yang lainnya *load file* KML yang kedua yang berisi *update* (kombinasi dari rubah, buat, dan hapus) ke dalam objek KML yang baru saja diload. *File* yang sudah terupdate berisi dua referensi untuk identifikasi data KML yang asli.
3. Untuk mengetahui lokasi objek dalam *Google Earth*, elemen update menggunakan elemen targetHref untuk mendefiniskan *file* asli yang didefiniskan sebagai objek untuk dimodifikasi. Untuk mengetahui *file* yang akan dimodifikasi, elemen rubah, buat, dan hapus terdapat elemen yang berisi atribut targetId yang mereferensikan identitas-identitas (i*ds*) dari objek tersebut.

.



Gambar 2.7. Mekanisme *Network link* [23]

Network Link A dan B merupakan *file* yang ada dalam *website* untuk melakukan pembaharuan data jika ada perubahan dalam *database.* Perubahan ini memberitahukan *file* asli yang telah di*load* oleh *Google Earth* untuk melakukan perubahan data pada tampilan *Google Earth*. Dalam beberapa kasus, network link dapat langsung diarahkan ke *file* sumber pada *website* yang menghasilkan *file* kml.

BAB 3

PERANCANGAN SISTEM PEMANTAUAN CUACA

Sistem yang akan di bangun terdiri atas dua bagian yaitu bagian sensor dan bagian aplikasi *website*, bagian sensor terdiri dari bagian piranti keras, piranti lunak, dan aplikasi *website*. Bagian piranti keras berfungsi untuk melakukan akuisisi data dari sensor GPS, kelembaban, dan suhu udara. Bagian piranti lunak berada pada sisi komputer dan berfungsi untuk mengatur penerimaan data dari sensor yang kemudian digunakan sebagai aplikasi untuk menulis ke pangkalan data *(database)* MySQL yang berada pada *server* dari sistem ini. *Server* dihubungkan dengan aplikasi PHP berinteraksi untuk menghasilkan data geografi berupa KML (*Keyhole MarkUp Language*) yang digunakan sebagai *file* utama pada *Google Earth*. Lewat mekanisme *Networklink* pada *Google Earth*, *file* KML tersebut dapat diperbaharui terus menerus sesuai dengan interval yang kita inginkan. Bagian dari sistemnya adalah sebagai berikut :

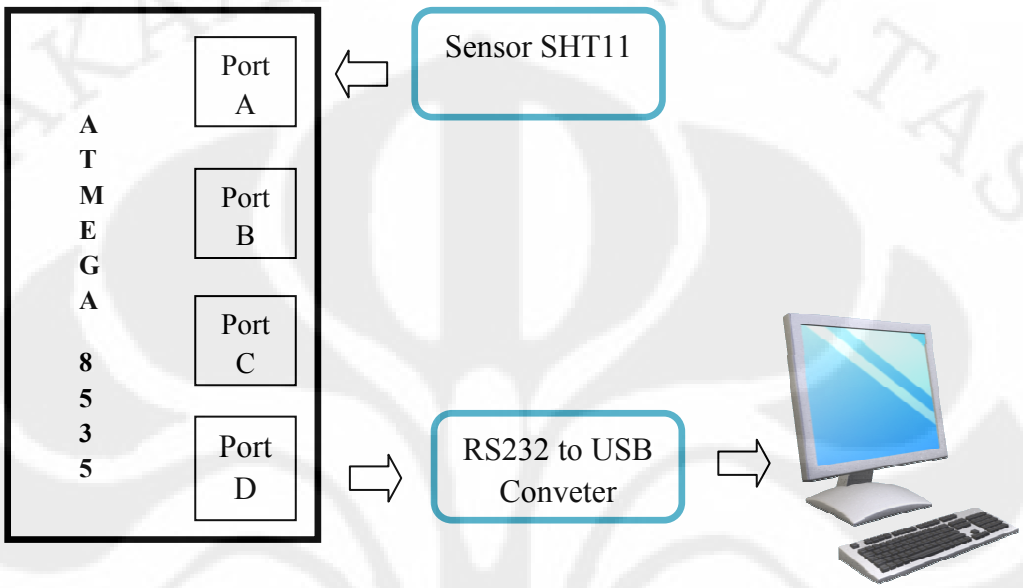
3.1 Bagian Sensor

3.1.1. Sensor SHT11

Sensor SHT11 dikonfigurasikan dengan menggunakan mikrontroler jenis AVR yaitu ATMEGA8535 yang berada pada sistem minimum DT-AVR Low Cost Micro System. Untuk melakukan antarmuka dengan komputer dengan menggunakan koneksi RS232 sehingga dapat dilakukan koneksi kedalam komputer.

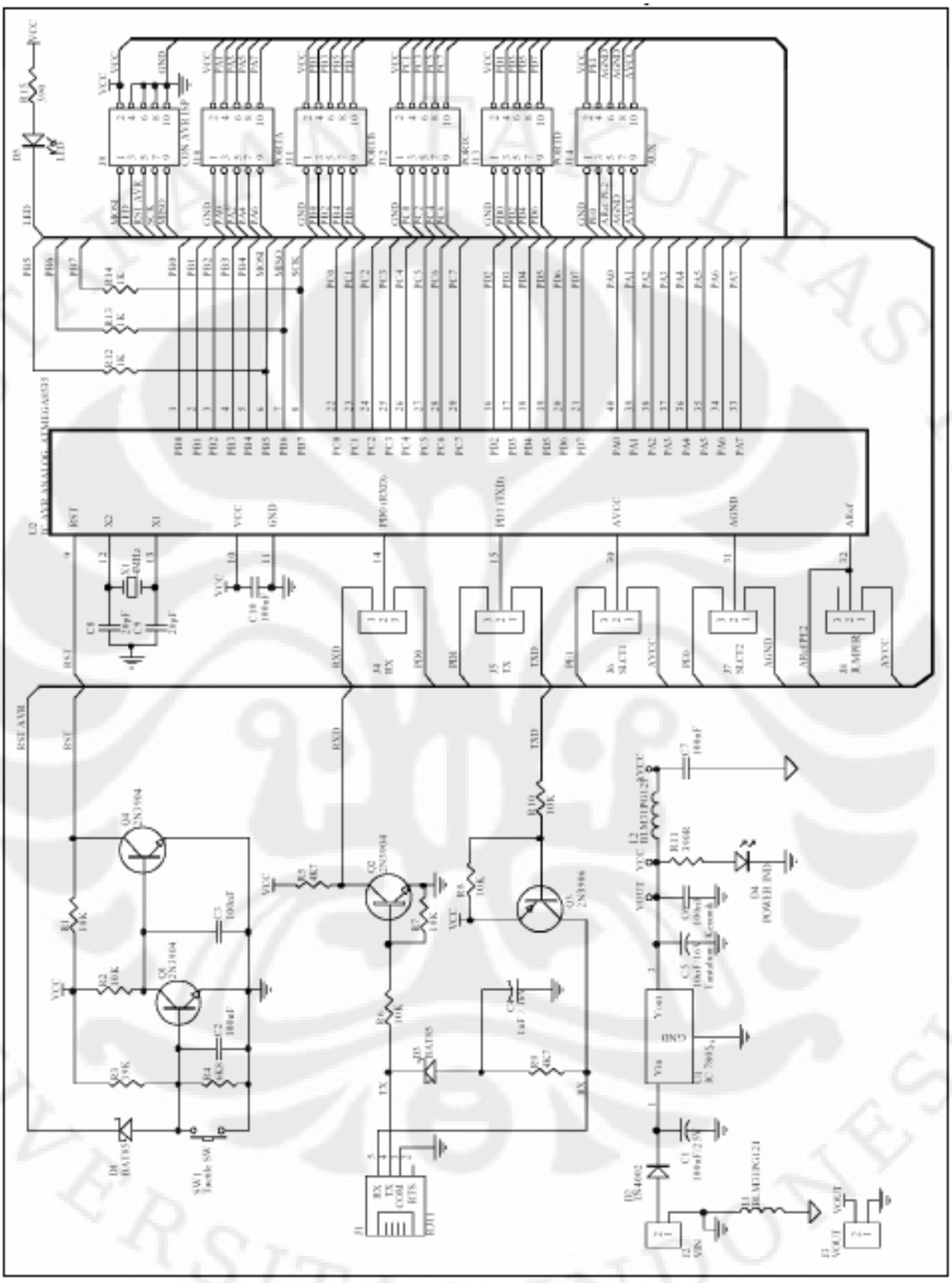
Untuk mendapatkan nilai pengukuran maka data diolah dengan menggunakan mikrokontroler sehingga nilainya dapat dikirimkan komputer melalui kabel serial RS232 untuk dapat diolah lebih lanjut. Pemrograman yang digunakan pada mikrokontroler adalah dengan C sedangkan pada komputer menggunakan lingkungan pemrogramanDelphi untuk melakukan pengolahan lebih lanjut.

Sensor SHT ini dikontrol oleh mikrokontroler jenis AVR 8535. Skema rangkaian perangkat kerasnya dapat dilihat pada gambar berikut ini :



Gambar 3. 1 Konfigurasi ATMEGA 8535 dengan SHT11

Berdasarkan gambar tersebut, sensor dibaca dari *Port* A pada mikrokontroler. Mikrokontroler membaca data bit yang langsung dapat di konversikan menjadi pembacaan suhu. Sensor ini sudah tidak perlu di kalibrasi karena perusahaannya sendiri, Sensirion Inc sudah melakukan kalibrasi sebelumnya. Dalam rancangan ini *Port* D merupakan data keluaran untuk melakukan komunikasi dengan dunia luar melalui koneksi RS232. Untuk melakukan komunikasi dengan PC, maka level tegangan TTL dari mikrokontroler (0-5 Volt) harus dikonversikan agar sama dengan level tegangan dari RS232 komputer. Untuk melakukan perubahan tegangan ini digunakan rangkaian MAX232. Dalam DT-AVR sendiri, rangakaian pengubah tegangan sudah ada dalam modul dengan menggunakan rangkaian 2 buah transistor. Rangkaian dari konfigurasi sistemnya adalah sebagai berikut :



Gambar 3. 2 Skema Sistem Minimum DTR-AVR

PA.0 DT-AVR Low Cost Micro System digunakan untuk membaca dan menulis dari dan ke SHT11. Sedangkan PA.1 DT-AVR Low Cost Micro System digunakan untuk menghasilkan pulsa (*clock*) untuk sinkronisasi proses komunikasi 2 *wire*. Hubungan antara kedua divais ini dapat dilihat pada tabel

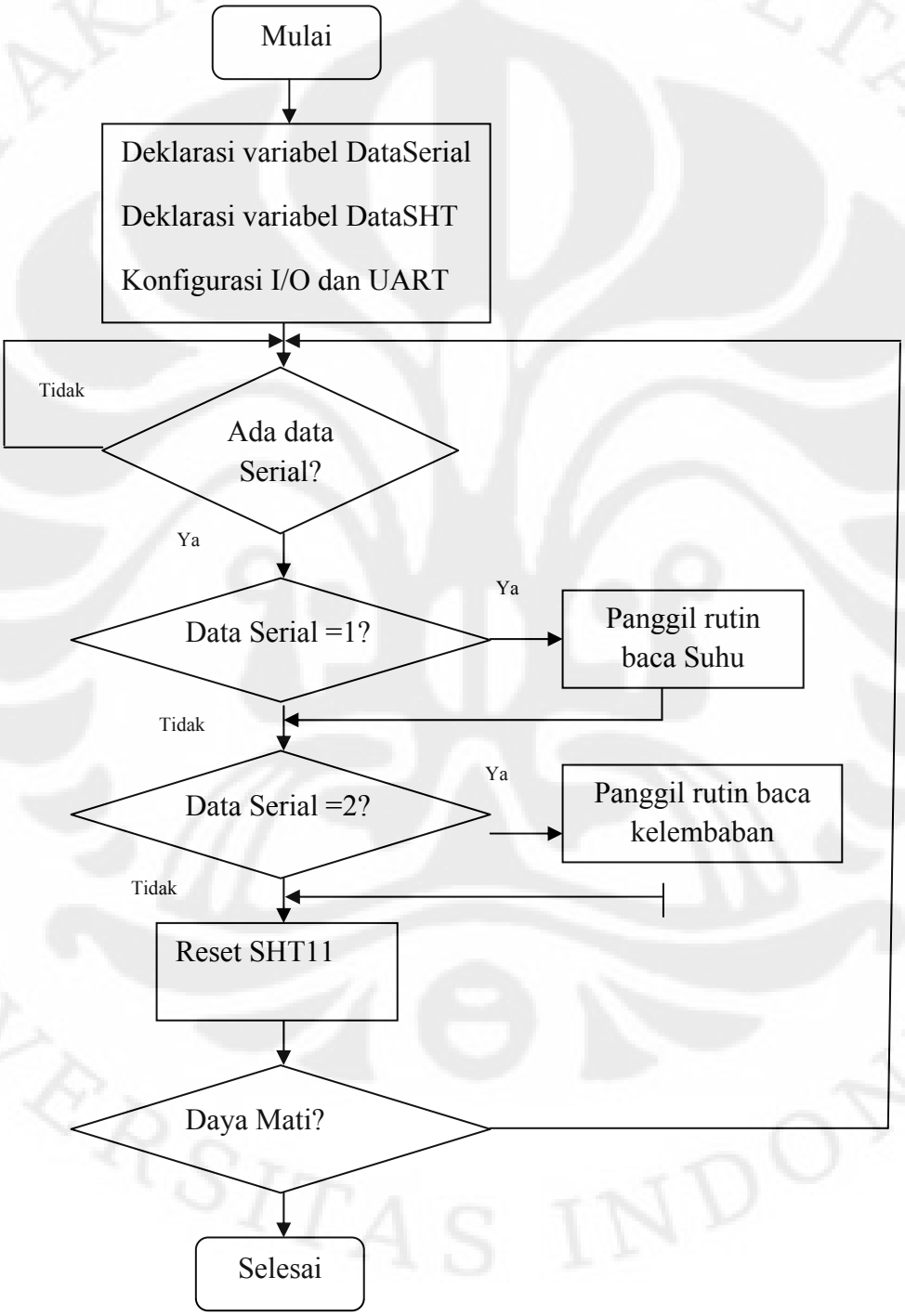
Tabel 3. 1 Tabel Koneksi DT-AVR dengan Modul SHT11

DT-AVR *Low Cost Micro System* (J10)	Modul SHT11
GND(Pin1)	*Ground* (Pin4)
VCC (Pin 2)	+5 VDC (Pin 8)
PA.0 (Pin 3)	Data (Pin 1)
PA.1 (Pin 4)	*Clock* (Pin 3)

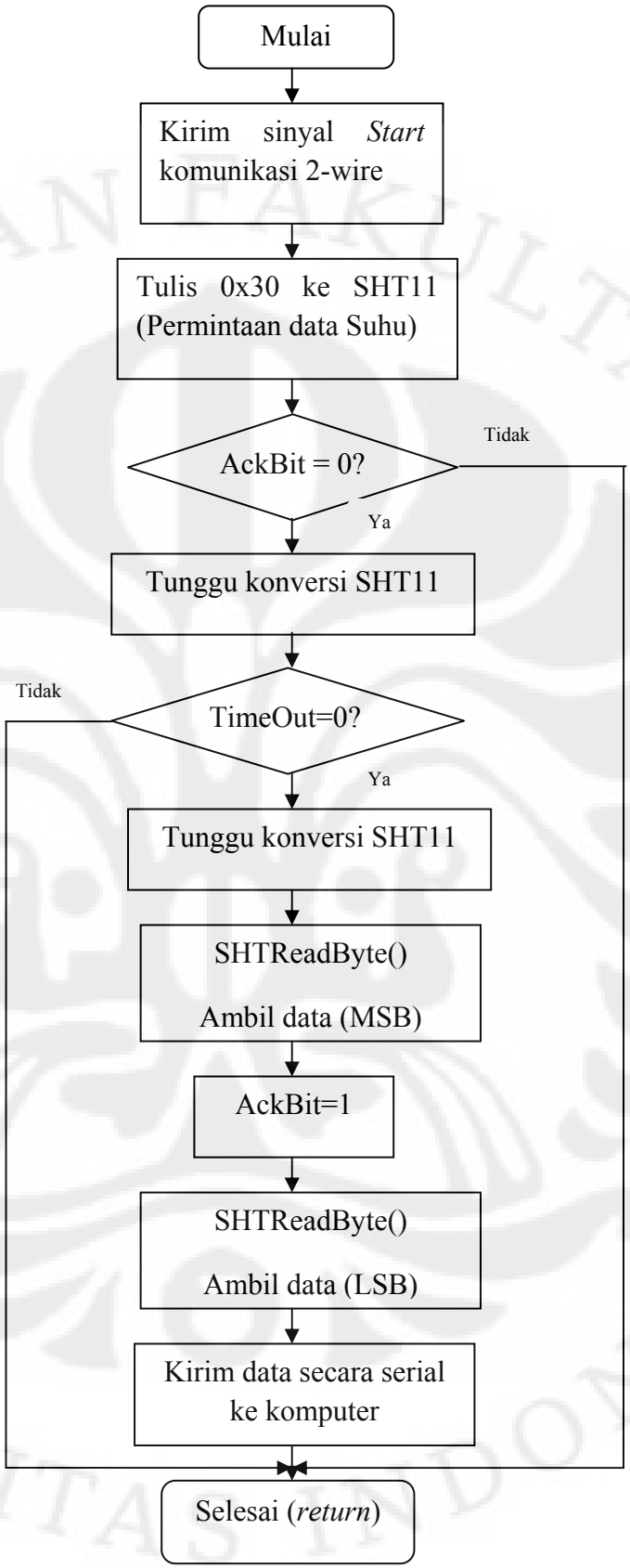
Diagram alir program utama mikrokontroler adalah dilihat pada gambar 3.3. Proses pertama yang dilakukan deklarasi variable DataSHT. Data serial adalah variabel yang digunakan untuk menampung data yang dikirim ke komputer. Data SHT11 adalah variabel yang menampung hasil pembacaan dari sensor SHT11. Program untuk membaca suhu digunakan rutin SHTReadTemp() dan SHTReadHumidity() secara umum adalah sama. Letak perbedaannya adalah nilai parameter pemanggilan fungsi SHTWriteByte(byte). Permintaan data suhu dilakukan dengan mengisi byte = 0x03 sedangkan permintaan kelembaban dilakukan dengan mengisi byte = 0x50.

Pada gambar 3.4 pembacaan suhu dimulai dengan mengirimkan sinyal *start* untuk memulai komunikasi 2-*wire*. Setelah itu program mengirimkan 0x30 ke SHT11 yang merupakan perintah untuk melakukan pengukuran suhu. Rutin SHTWriteByte(0x30) akan memberikan nilai ACK yang disimpan dalam variabel AckBit. Jika variabel AckBit bernilai 0, maka program akan menunggu selesainya pengukuran SHT11 dengan memanggil rutin SHTWait(). Rutin SHTWait() akan memberikan suatu nilai yang kemudian disimpan pada variabel TimeOut. Variabel TimeOut akan bernilai 0 jika pengukuran SHT11 selesai dan data siap.

Setelah pengukuran selesai, data suhu akan dibaca LSB dulu kemudian MSB. Pembacaan data LSB dilakukan dengan memberi nilai variabel AckBit=0, sedangkan pembacaan data MSB dilakukan dengan memberi nilai AckBit=1. Data hasil pembacaan suhu kemudian dikirimkan secara serial ke komputer



Gambar 3. 3 Diagram Alir Program Pada Mikrokontroler



Gambar 3. 4 Diagram Alir untuk Membaca Suhu

Program pada sisi komputer dikembangkan dengan Borland™ Delphi 7.0© dan komponen C*Port*. Jika Program mengirimkan karakter '1' maka dilakukan pemanggilan data suhu dan jika karate '2' yang dikirim, maka akan dilakukan pemanggilan data kelembaban. Program ini dilakukan secara berulang antara suhu dan kelembaban.

3.1.2 Bagian Sensor Posisi (GPS)

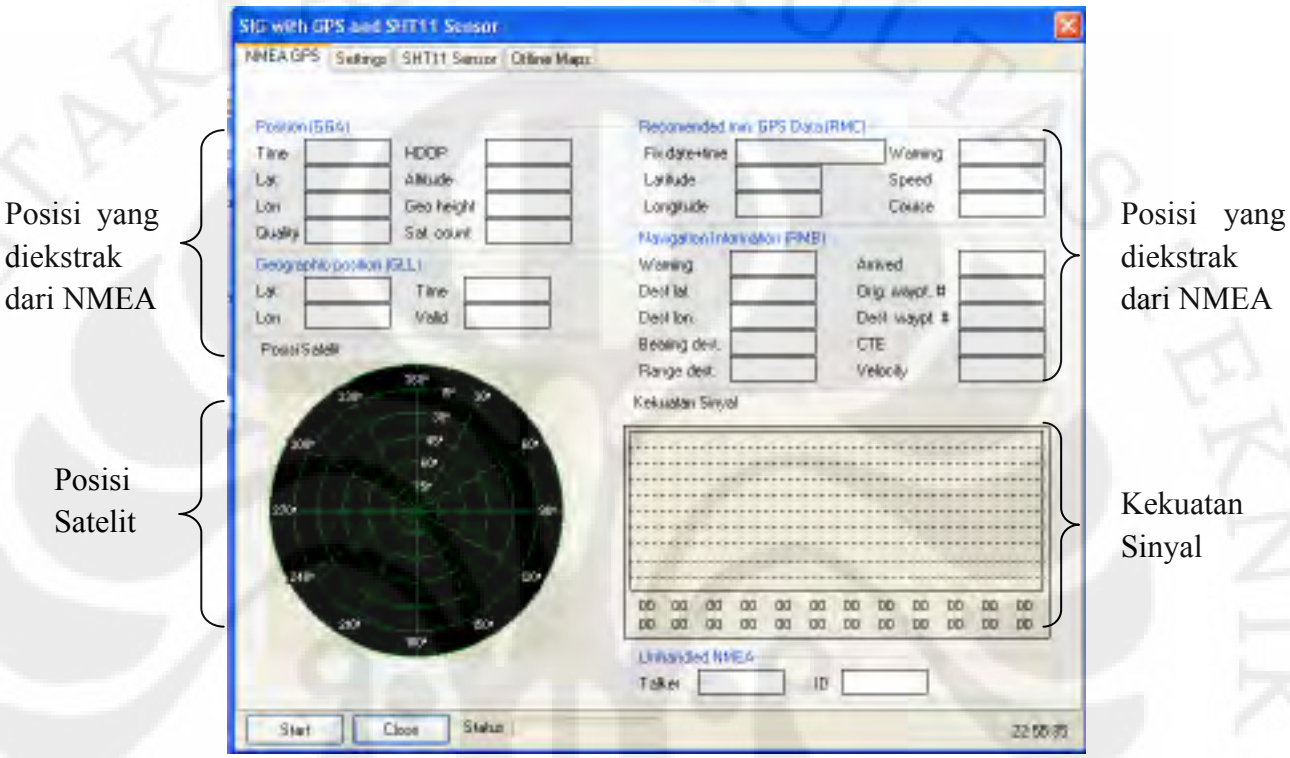
Konfigurasi yang dipilih adalah menggabungkan langsung GPS ke dalam komputer melalui kabel USB. GPS yang dipilih adalah jenis Garmin PC 18 USB. Untuk mendapatkan keluaran NMEA 0813 maka digunakan piranti *Spanner* (GPS GATE) karena keluaran dari Garmin PC 18 USB adalah standar garmin. Konfigurasi ini dapat diketahui secara aktif mengeluarkan informasi posisi dapat dilihat langsung dalam *Hyperterminal* dengan sebelumnya melakukan konfigurasi dengan menggunakan *baudrate* 9600 maka didapatkan data lokasi.

Gambar 3. 5 Garmin PC 18 USB

3.2 Lingkungan Pemrograman Delphi

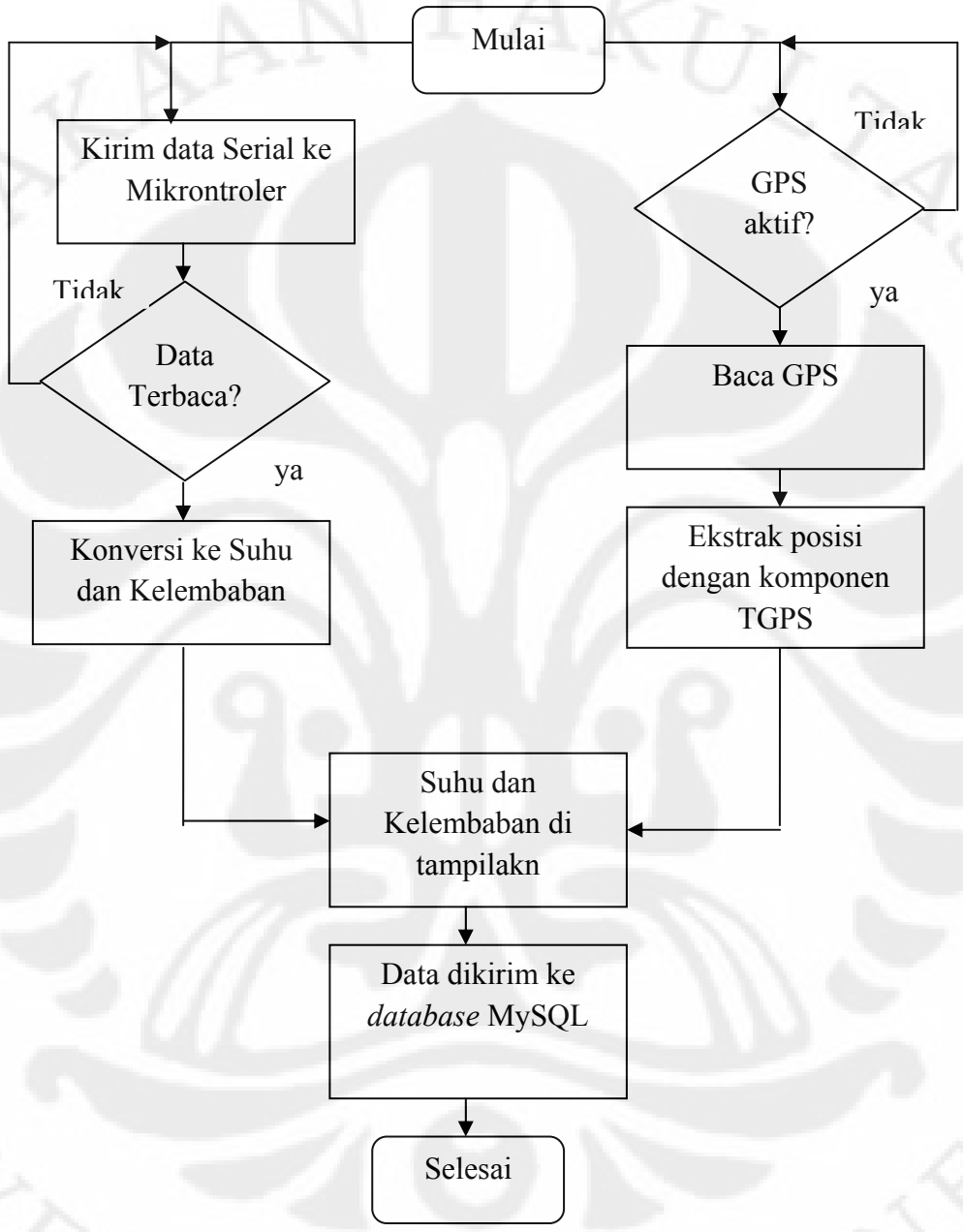
Lingkungan Pemrograman Delphi dipilih karena mudah untuk diintegrasikan dalam sistem tertanam (*embedded*). Selain itu lingkungan pemrogramanini menawarkan komponen-komponen yang bisa langsung

digunakan untuk sistem informasi geografi seperti TGPS. Komponen ini nantinya digunakan dalam pemrograman untuk membuat aplikasi *desktop*.



Gambar 3. 6 Piranti Lunak pada Komputer Kontrol

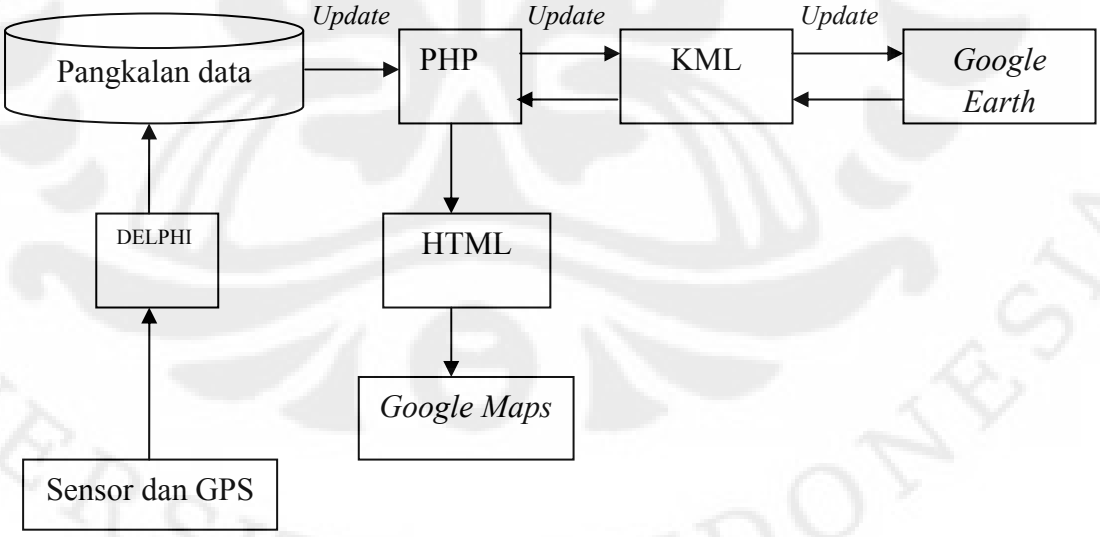
Lingkungan pemrogramanDelphi digunakan untuk membuat piranti lunak ini digunakan pada sisi komputer ini digunakan sebagai pengumpul data dari sensor yang dikirimkan secara serial melalui RS232. Data yang diperoleh dari sensor kemudian diolah menjadi data yang siap untuk dikirimkan kembali ke dalam pangkalan data *(database)* MySQL melalui komponen Zeos. Diagram Alir program pada *desktop* dapat dilihat pada gambar 3.6. Seperti penjelasan sebelumnya sistem dapat membaca suhu dan kelembaban dengan mengirimkan data serial ‘1’ dan ‘2’. Data kemudian ditangkap dan dikonversikan terlebih dahulu sebelum memperoleh nilai yang diinginkan. Sedangkan nilai GPS di peroleh secara langsung dengan komponen TGPS yang ada dalam Delphi 7.0. Bersama dengan data SHT11 dan GPS di tampilkan dan kemudian secara teratur di simpan dalam *server* MySQL untuk dapat diolah lebih lanjut.



Gambar 3. 7 Diagram Alir pada Program Pengolah di Komputer Kontrol

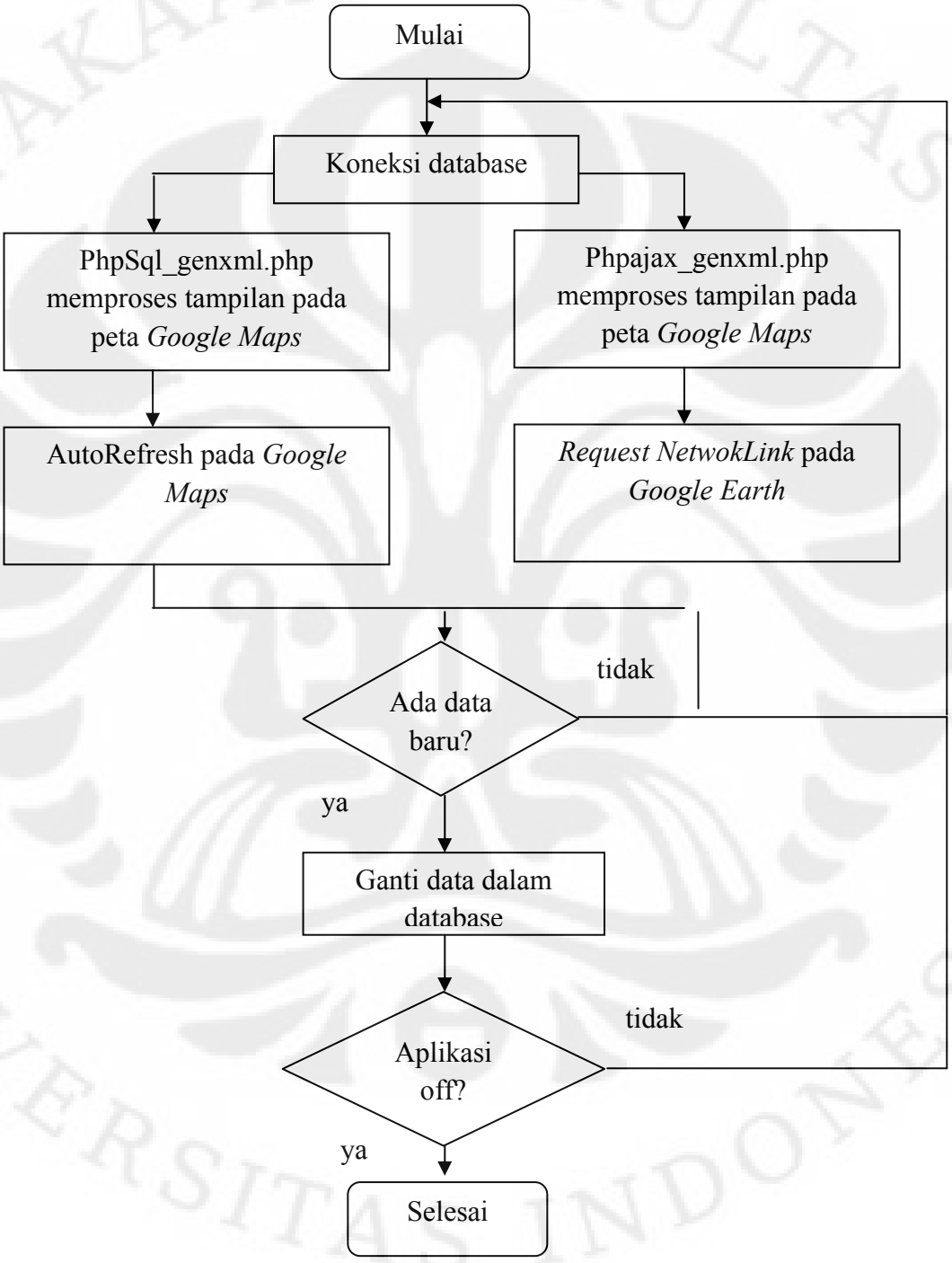
3.3. Aplikasi *Website*, KML, dan *Google Earth*

Aplikasi *website* yang akan dibuat adalah dengan menggabungkan data yang didapatkan dari pengukuran posisi dan lokasi dengan menggunakan GPS dan data suhu yang ada. *Google Earth* digunakan sebagai penampil data dengan terlebih dahulu membuka *file* kml yang telah dibuat oleh aplikasi PHP dengan data yang berasal dari MySQL. Data sensor yang diperoleh dikumpulkan dalam komputer dengan interval waktu tertentu kemudian diolah menjadi pangkalan data *(database)* MySQL melalui program yang dibuat dalam lingkungan pemrogramanDelphi. Pangkalan data *(database)*nya disimpan dalam bentuk MySQL. *File* KML yang berisi informasi dihasilkan dari aplikasi *web* yang berbasis PHP dengan data yang berasal dari pangkalan data *(database)* MySQL. Data ini dapat di*unduh (download)* dalam *website server* yaitu lokal yang sudah konfigurasikan. KML ini nantinya digunakan sebagai *file* sumber dari *Google Earth* untuk melakukan penampakan informasi. KLM ini melalui fungsi *Networklink* dalam *Google Earth* sehingga data yang muncul merupakan data terbaru. Proses ini digambarkan secara lengkap pada gambar 3.8.



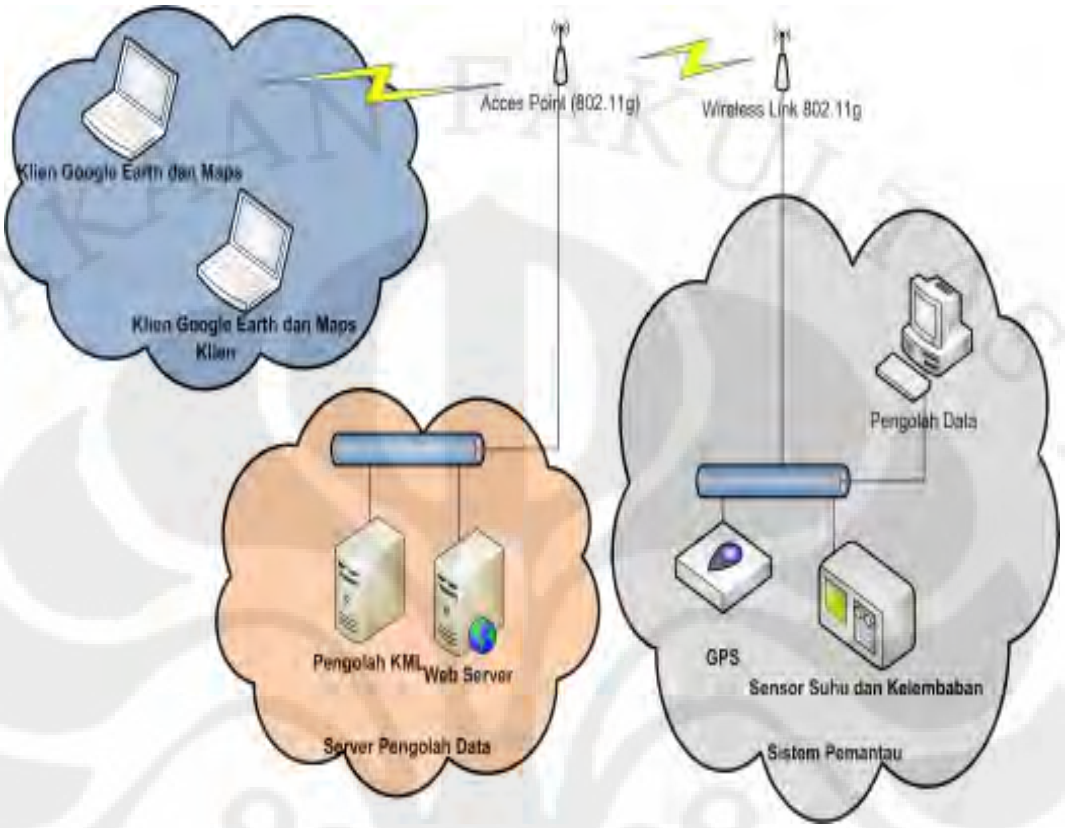
Gambar 3. 8 Sistem Pengolahan Database

Diagram alir pada aplikasi *web*site dapat terlihat pada gambar 3.9. Aplikasi *web*site mempunyai dua *file* utama yaitu phpsql_genkml.php/pantau.php/pantau semua.php yang memproduksi KML dan phpsqlajax_genxml.php yang menghasilkan posisi pada *Google Maps*.



Gambar 3. 9 Diagram Alir pada Aplikasi *Google Maps* dan *Google Earth*

3.4 Sistem Keseluruhan



Gambar 3. 10 Rancangan Sistem Pemantauan

Gambar 3.10 menunjukan sistem keseluruhan yang diajukan dalam skripsi ini dengan menggunakan Acces Point 802.11g sebagai representasi dari komunikasi satelit. Sistem ini terdiri dari tiga bagian utama yaitu bagian cuaca (sistem pemantau), server, dan *client*. Penjelasan masing-masing komponen sebagai berikut

3.4.1 Sistem Pemantau

Sistem pemantau ini terdiri atas GPS dan Sensor SHT11 yang berada dalam sistem pemantau dengan koneksi internet satelit. Dalam skripsi ini sistem pemantau menggunakan laptop sebagai pengolah data dan koneksi internet 3G Indosat M2 dan Wireless 802.11g dan pergerakan di lakukan di sekitar Universitas Indonesia dengan menggunakan kendaraan bermotor

3.4.2 *Server*

Server dalam sistem ini menggunakan komputer. *Server* ini menerima data dari aplikasi untuk pengolahan data sensor. *Server* ini menyimpan sistem dalam tabel tertentu dengan urutan tertentu untuk kemudian ditampilkan dalam *Google Earth* dan *Google Maps*

3.4.3 *Client*

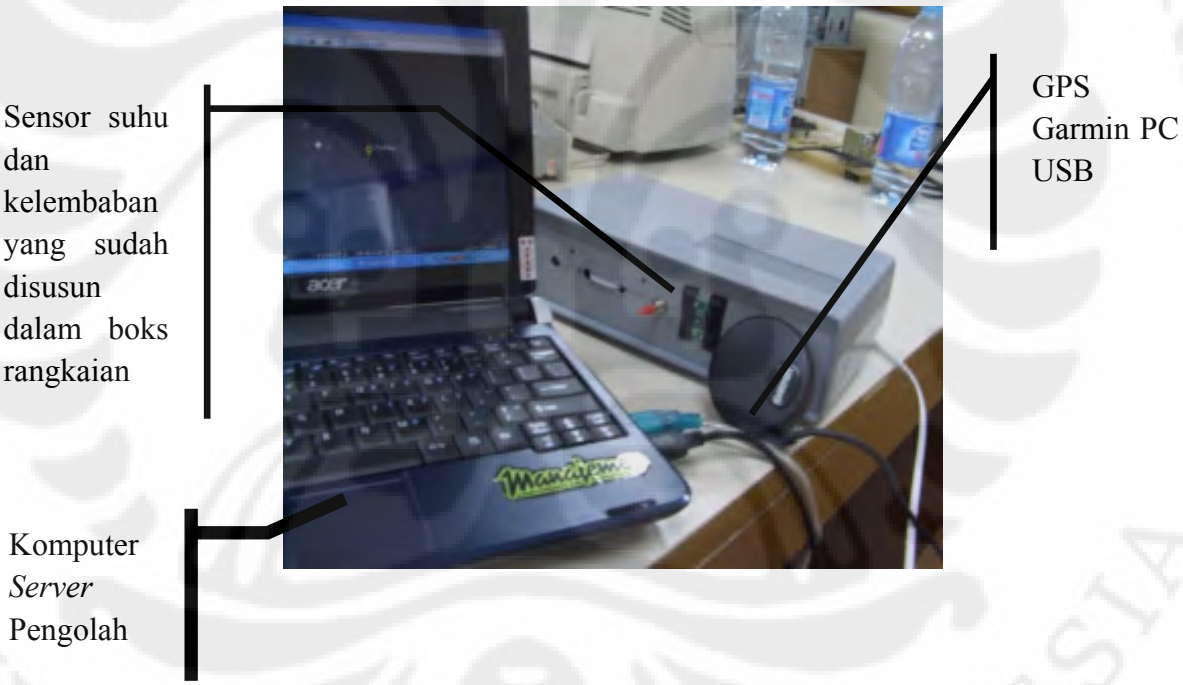
Client adalah pihak atau lembaga yang menggunakan data pemantauan dengan *Google Earth* dan *Google Maps*. Dalam komputer *client* harus mempunyai koneksi internet dan ter*install* piranti lunak *Google Earth* serta *web browser* seperti Firefox, Internet Explorer, dan Opera.

BAB 4

ANALISIS KINERJA SISTEM PEMANTAUAN CUACA

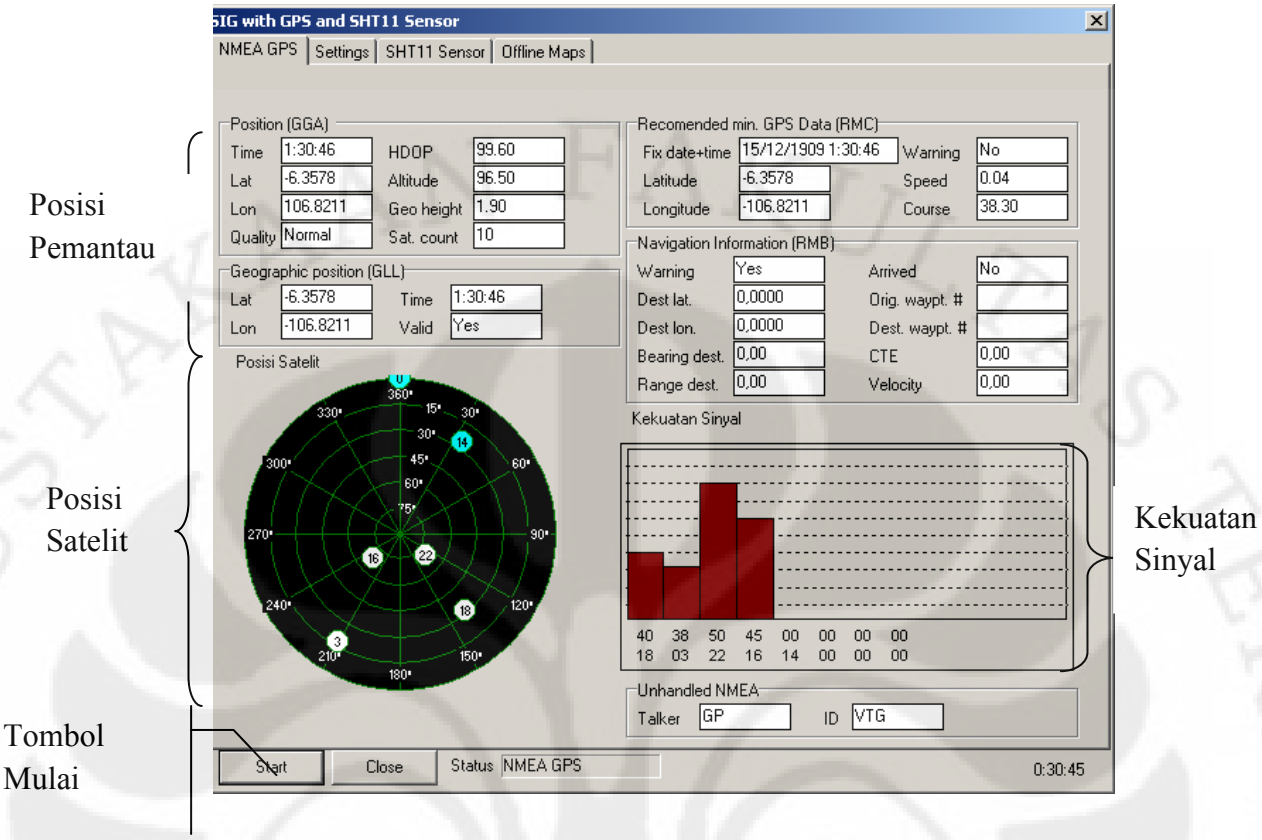
4.1 Prototype Sistem Pemantauan Cuaca

Sistem pemantauan cuaca ini berbasis komputer sebagai pengolah data. Pengolah data bisa dikembangkan di kemudian hari dengan menggunakan PC Embedded Semisal EBOX 4300 yang berbasis *Windows CE*. Sistem ini menggunakan terdiri atas dua penampilan data yaitu pada *Google Earth* dan *Google Maps*. Data yang ditampilkan antara lain posisi lintang dan bujur, kecepatan, suhu, kelembaban, *dewpoint*, dan parameter cuaca laut (ada tidaknya kabut). Sistem *Prototype* Pemantauan Cuacanya sebagai berikut :



Gambar 4. 1 *Prototype* (Purwarupa) Sistem Hasil Rancangan

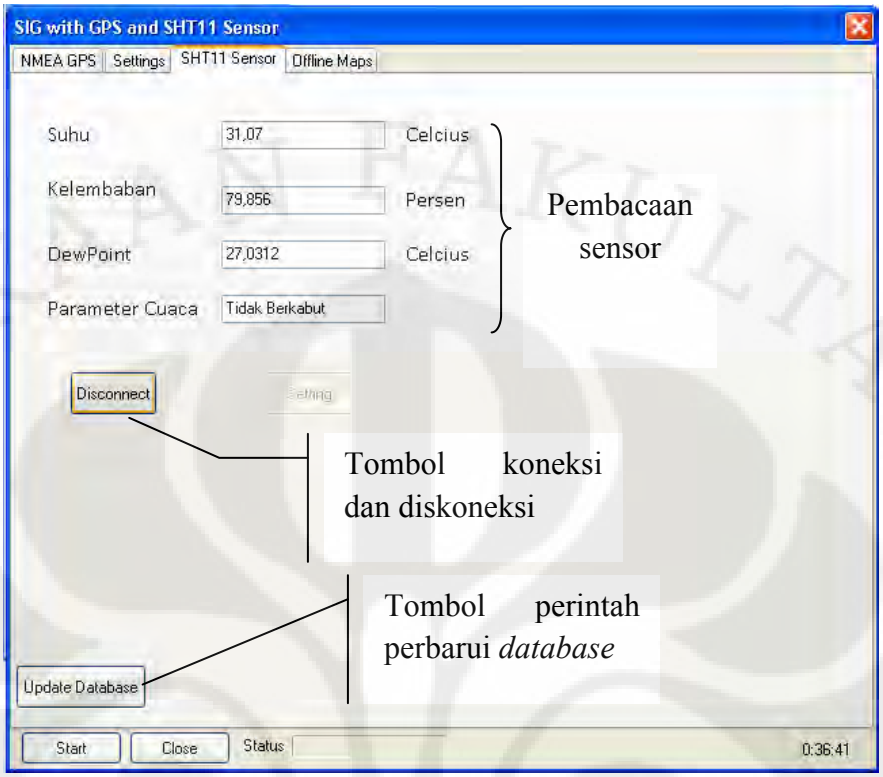
Sistem ini menggunakan spesifikasi Komputer Core Duo Merk HP 520 dengan RAM 1 GB dan *Hardisk* 80 GB sebagai *server* dan pengolah aplikasi sedangkan GPS yang digunakan adalah GPS USB 18 dan Sensor yang telah dirancang dalam suatu box sehingga lebih ringkas.



Gambar 4. 2 Antar muka Aplikasi GPS pada Komputer

Gambar 4.2 menunjukan aplikasi antar muka GPS pada aplikasi di komputer. Aplikasi ini menggunakan dibuat dengan lingkungan pemrogramanDelphi 7.0 dengan memanfaatkan komponen TGPS. Ada 3 jenis data komponen posisi yang diperoleh yaitu yang diekstrak dari GGA, GGL, dan GPRMC yang menghasilkan komponen garis bujur dan garis lintang. Garis bujur dan lintang yang diperoleh untuk dimasukan ke dalam *database* adalah dari kalimat GGL. Komponen TGPS juga mempunyai kemampuan untuk melakukan *tracking* satelit dan kekuatan sinyal penerimaan GPS.

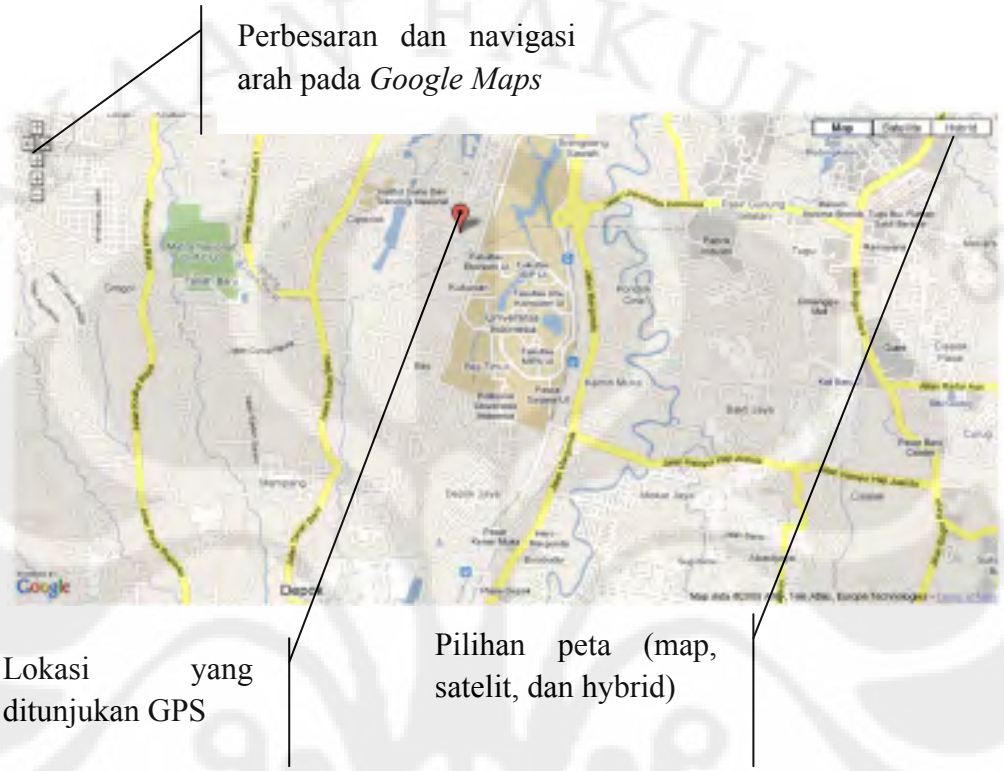
Gambar 4.3 menunjukan aplikasi antar muka untuk pemantauan suhu, kelembaban, dewpoint, dan parameter cuaca laut. Bagian aplikasi ini memberikan informasi tentang parameter cuaca permukaan laut. Data sensor diperolah dari komunikasi serial dengan mikrokontroler.



Gambar 4. 3 Antarmuka Aplikasi Sensor SHT11 pada Komputer

Aplikasi pada *Google Maps* terdiri dari sebuah *ballon* yang menggambarkan lokasi dan juga aplikasi *Google Maps* yang menampilkan data lokasi secara *real time*. Sistem antarmuka *web* menggunakan fungsi *iframe* yang membagi *browser* menjadi beberapa bagian sehingga perbaharuan data pada lokasi tidak mengganggu perubahan data pada bagian yang lain

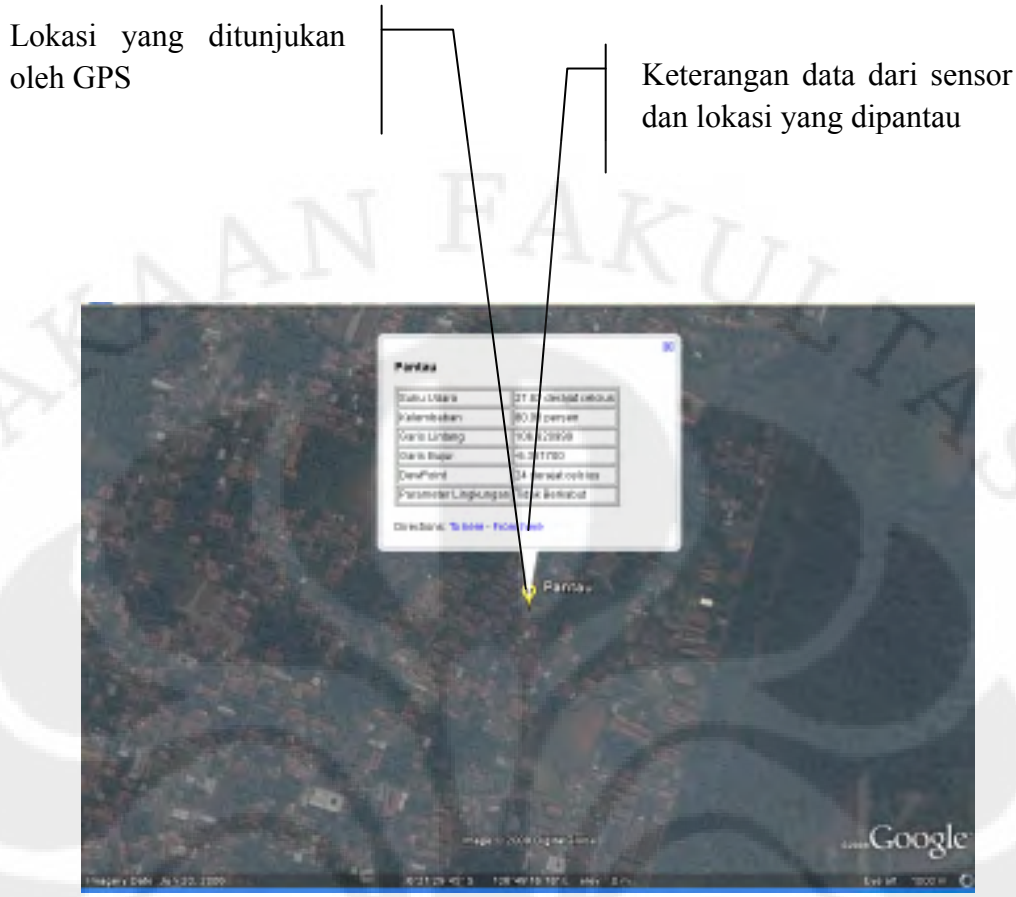
Aplikasi *Google Maps*nya sebagai berikut



Gambar 4. 4 Aplikasi *Google Maps*

Gambar 4.4 menggambarkan aplikasi *Google Maps* untuk penentuan lokasi berdasarkan lokasi yang ditunjukan oleh GPS. Pembacaan yang dilakukan oleh sensor suhu dan kelembaban di tampilkan di halaman lain dari *web*site dengan menggunakan fungsi pemanggilan dari php. Aplikasi ini hanya mencakup posisi saja dan tidak menyertakan keterangan suhu dan kelembaban.

Gambar 4.5 merupakan aplikasi *Google Earth* menggunakan *Google Earth* versi 5 tidak berbayar dengan yang sebelumnya telah di *install* di komputer. Data pada *Google Earth* berupa lokasi dan keadaan cuaca yang dapat terlihat dari *ballon* ketika *pushpin* pada layar *Google Earth* di klik. Tampilan keterangan yang muncul adalah garis bujur, garis lintang, suhu, kelembaban, *dewpoint*, parameter cuaca, dan waktu. Penampilan lokasi (*pushpin*) terjadi karena aplikasi ini memperoleh data dari *server* MySQL melalui fungsi *networklink* dari *Google Earth.*

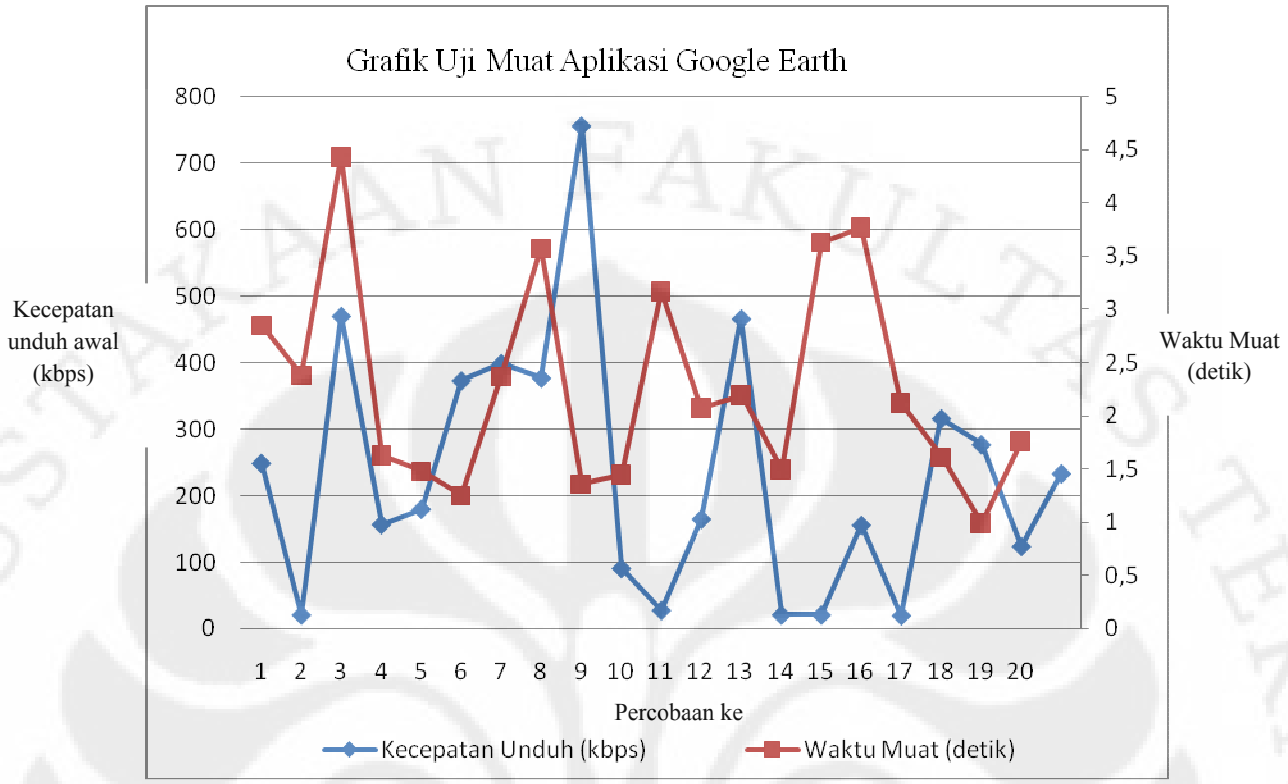


Gambar 4. 5 Aplikasi *Google Earth*

4.2 Uji Waktu Muat (*Loading Time*)

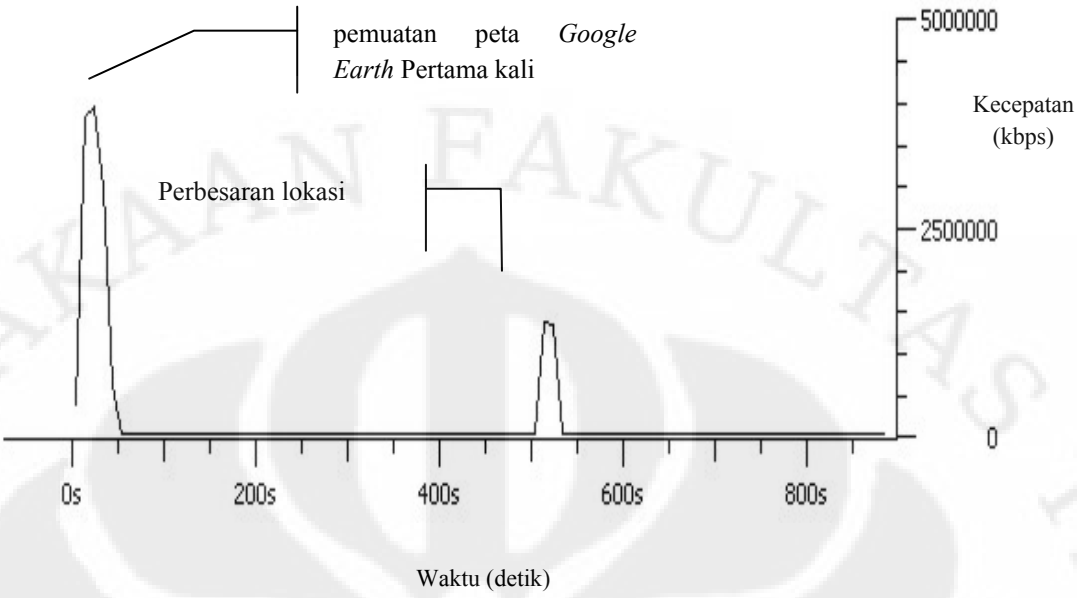
4.2.1. Uji Muat Waktu Muat Aplikasi *Google Earth*

Uji waktu muat digunakan untuk mengetahui besarnya waktu muat dari sistem yang diberi masukan data konvensional (bukan dari data sensor), namun data dapat dikatakan mewakili data yang didapatkan dari sensor. Data yang diambil adalah interval antara waktu pemasukan data sampai dengan aplikasi tersebut berjalan dengan memvariasikan waktu percobaan Grafik dari uji muat aplikasi adalah sebagai berikut



Gambar 4. 6 Grafik Uji Muat Aplikasi *Google Earth*

Berdasarkan perbandingan yang ada antara uji muat dan kecepatan unduh *(download)*. Rata-rata kecepatan muat *Google Earth* pada aplikasi ini sekitar 2,28 detik dan bervariasi terhadap kecepatan unduh (pengolahan data pada Lampiran). Pada saat aplikasi dijalankan terlihat bahwa hampir tidak ada hubungan antara kecepatan unduh *(download)* awal dengan kecepatan uji waktu. Hal ini terjadi karena aplikasi *Google Earth* merupakan program komputer yang sekaligus merupakan aplikasi internet sehingga untuk mendapatkan waktu uji muat yang lebih baik maka performa komputer dan juga internet harus lebih baik. Ketidakteraturan dalam melakukan pengunduhan *(download)* ini dapat dijelaskan pada grafik pada lampiran. Gambar 4.7 menunjukan bahwa sistem melakukan pemanggilan paket data berupa peta pada saat terjadi pemanggilan peta pertama kali.

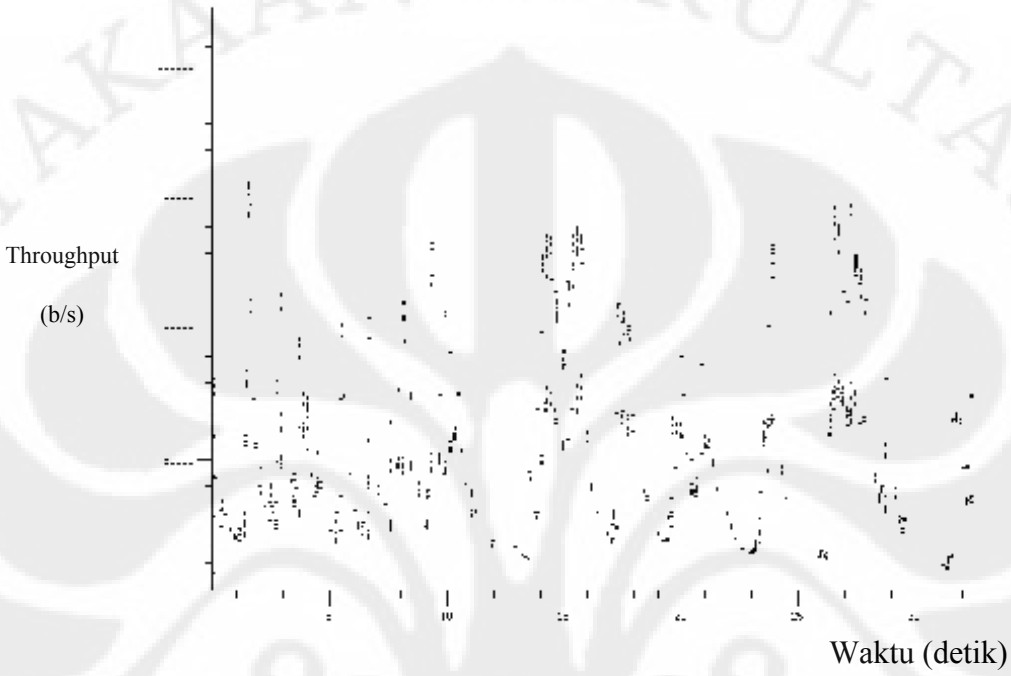


Gambar 4. 7 Grafik TCP IP IO Graph pada *Wireshark*

Dari gambar 4.7 yang dihasilkan dari maka terlihat bahwa penggunaan *Google Earth* termasuk hemat dalam pemakaian sumber daya internet. Pada grafik terlihat bahwa nilai maksimal untuk melakukan inisialisasi aplikasi adalah 400 kbps (dilihat dari tabel yang sebelumnya dibagi dengan 10) sedangkan untuk melakukan perbaharuan informasi setiap 20 detik hampir tidak membutuhkan sumber daya internet seperti pada *Google Maps*. Pemuatan peta pun hanya dilakukan sekali saja pada saat awal dan pemuatan ini berjalan sekitar 40 detik. Hal ini terjadi karena mekanisme perbaharuan data pada *Google Earth* tidak mengambil data keseluruhan lagi, namun dengan menggunakan mekanisme *Networklink* yang ada *Google Earth* sehingga dapat menghemat sumber daya internet. Pada detik ke 500-550 terjadi perbesaran lokasi. Perbesaran lokasi ini ternyata membutuhkan sumber daya internet untuk dapat melakukan perbesaran lokasi.

Pada grafik *Troughput* ,terlihat bahwa penggunaan data pada *Google Earth* hanya sampai detik ke 40 yang artinya sumber yang dominan dalam melakukan perbaharuan informasi adalah *server* penyimpan data sensor dan lokasi yang dalam hal ini adalah XAMPP yang ada di komputer (*localhost*) sehingga tidak memerlukan sumber internet kecuali jika terjadi perpindahan lokasi yang

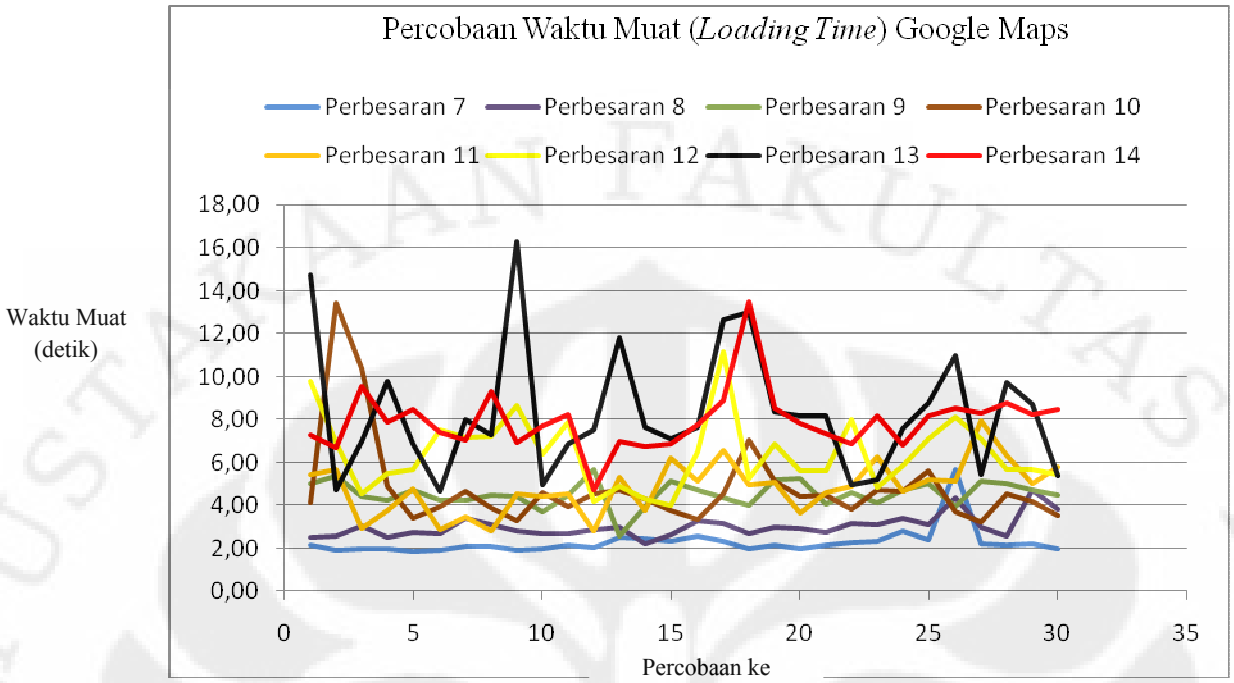
signifikan, maka *Google Earth* akan memperbaharui tampilan petanya. Namun hal ini secara nyata tidak akan terjadi karena tidak mungkin terjadi perpindahan geografis cuaca secara signifikan dalam jarak.



Gambar 4. 8 Grafik *Throughput* Aplikasi *Google Earth*

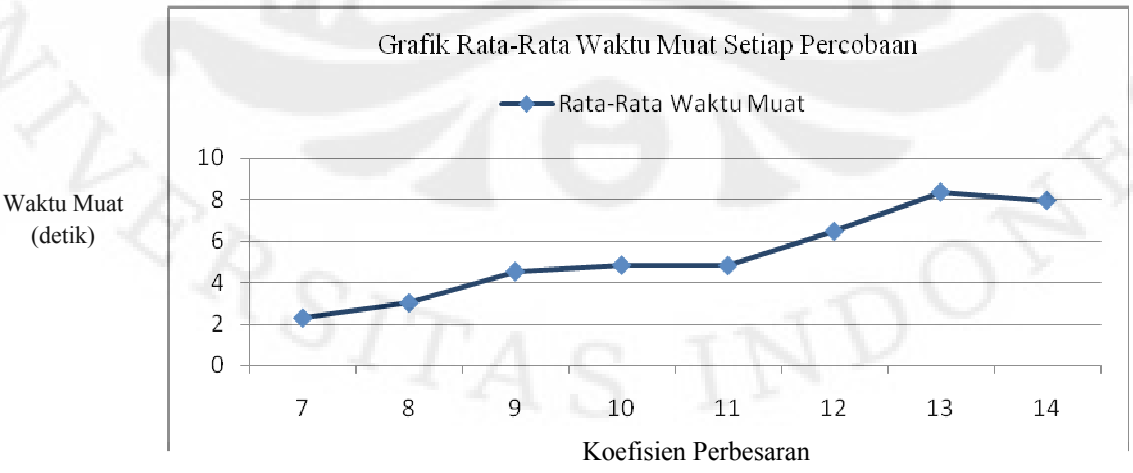
4.2.2. Uji Muat Waktu Aplikasi *Google* Maps

Uji waktu muat ini menggunakan *add-ons* dari Mozilla Firefox yaitu *Firebug*. Waktu uji muat divariasikan terhadap level pembesaran yang ada di *Google Maps* untuk mengetahui variasi waktu yang terjadi dengan menggunakan *add ons* (pengaya) *firebug* yang merupakan bagian dari aplikasi Mozille Firefox 3.6 beta 3. Metode selancar (*browsing*) yang digunakan adalah menggunakan metode *Private Browsing* adalah cara selancar (*browsing*) dimana data dari selancar sebelumnya seperti *cookies* dan *cache* tidak akan disimpan ketika melakukan pemuatan dari halaman baru.



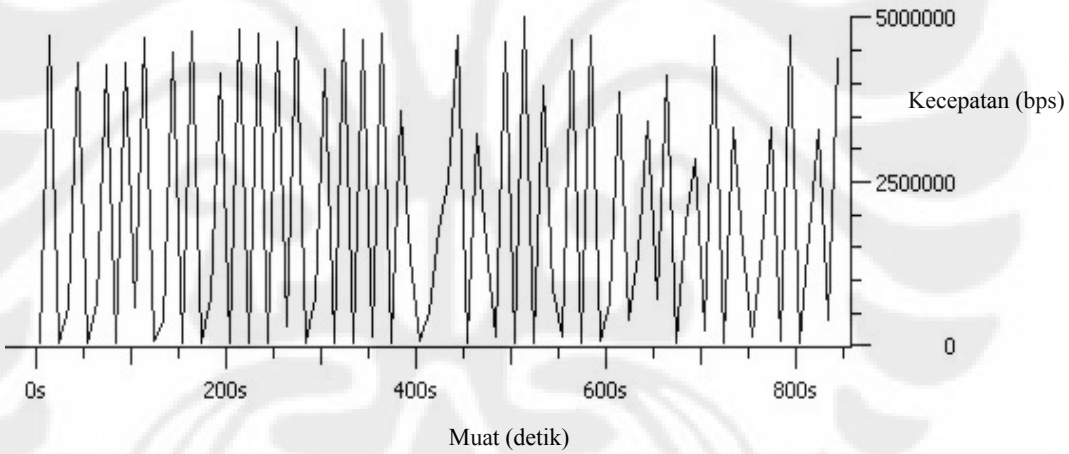
Gambar 4. 9 Percobaan Waktu Muat (*Loading Time*) *Google Maps*

Terlihat dari gambar 4.9 bahwa perbesaran dengan koefisien 13 dan 14 mempunyai kecepatan waktu muat yang relatif lebih lama dengan kecepatan *unduh (download)* dari internet yang hampir sama. Hal ini mengindikasikan bahwa pada *Google Maps* semakin dekat dengan objek yang akan di pantau, dalam percobaan ini menggunakan wilayah kampus UI Depok, maka semakin lama waktu muat yang digunakan.



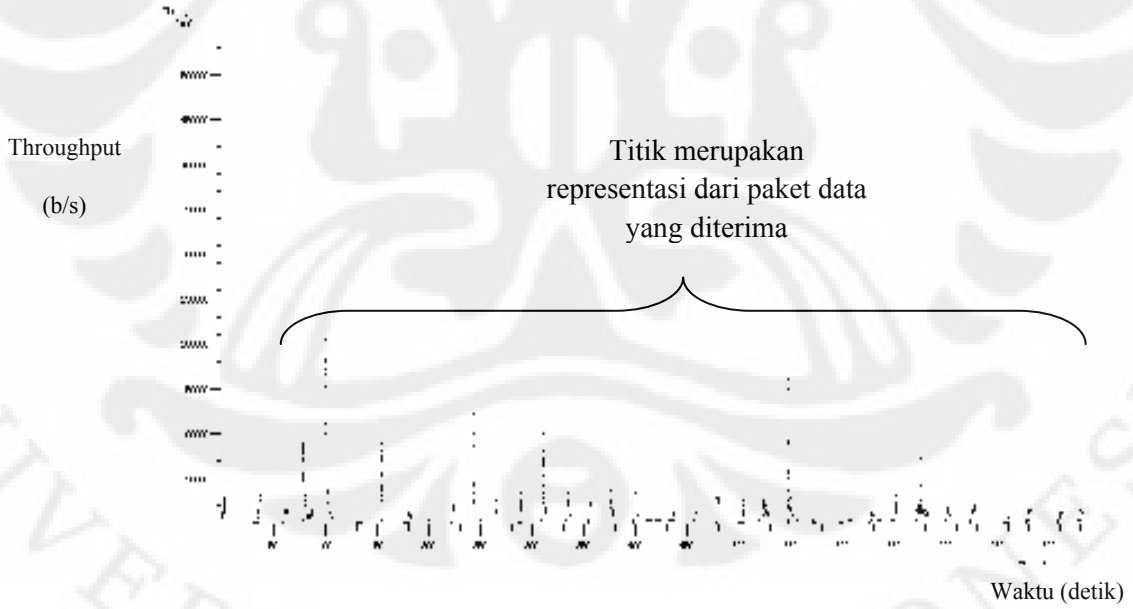
Gambar 4. 10 Grafik Rata-Rata Waktu Muat Setiap Percobaan

Rata-rata waktu muat untuk tiap koefisien perbesaran terlihat pada gambar 4.10. Koefiesen 7 dan 8 memiliki nilai waktu muat yang cepat karena data yang diambil tidak terlalu detail sehingga waktu untuk memuat data akan semakin cepat. Dari analisis ini dapat diambil kesimpulan untuk membuat waktu pembaharuan data dari *Google Maps*. Untuk daerah objek kawasan UI depok dapat digunakan waktu pembaharuan data sekitar 8 detik (harus diatas 7 detik) dengan asumsi bahwa kecepataan unduh *(download)* data adalah sama. Untuk kecepatan internet yang lebih rendah maka perlu diatur waktu pemuatan yang lebih lama karena untuk mengunduh *(download)* semua komponen gambar. Pengaturan waktu pemuatan peta ini sangat penting karena aplikasi *Google Maps* untuk mengubah kondisi suatu data yang ada maka harus diperbaharui seluruh peta yang artinya seperti mengambil suatu halaman baru.



Gambar 4. 11 Grafik TCP IO dari *Google Maps*

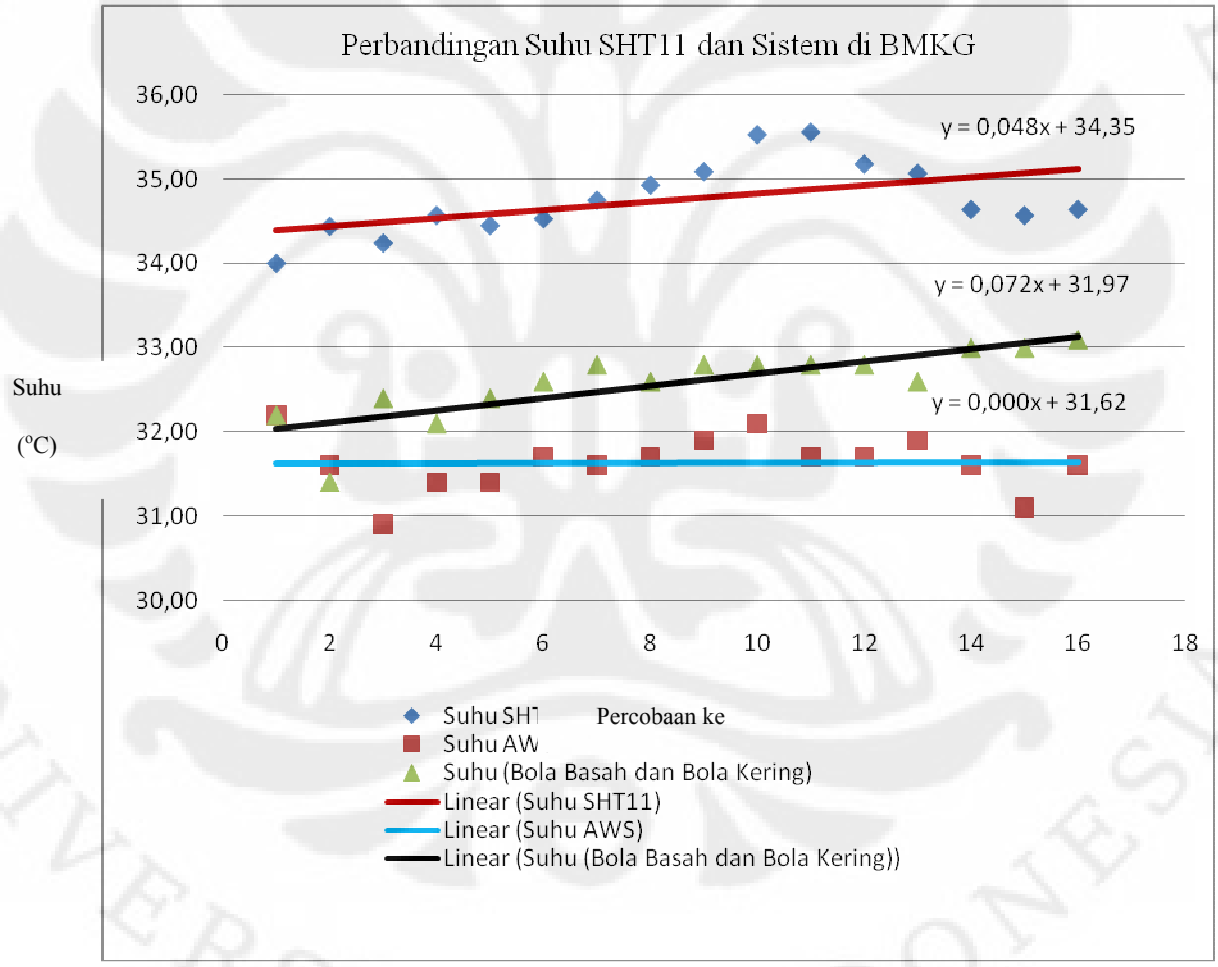
Gambar 4.11 merupakan hasil uji lalu lintas data dengan menggunakan *Wireshark*. Grafik di atas divariasikan terhadap waktu dalam detik dengan interval pembaruan halaman *website* sebesar 20 detik. Terlihat pada grafik bahwa terdapat kenaikan *bandwith* pada setiap interval 20 detik. Ini menunjukan bahwa terjadi pemuatan halaman baru di *Google Maps* dengan fungsi *autorefresh*. Setelah dihitung paket tertinggi mempunyai kecepatan unduh *(download)* paket datanya adalah berkisar 275 – 500 kbps. Semakin rendah kecepatan unduh *(download)* maka distribusi paket akan melebar sedangkan jika kecepatan unduh *(download)* semakin tinggi maka grafik cenderung runcing. Ini terjadi karena dengan memperbaharui halaman yang sama dan besar data yang sama serta karakteristik dari TCP yang akan mentransmisikan ulang data yang gagal karena koneksi. Grafik ini juga mempunyai kesamaan dengan grafik *Throughput* pada gambar 4.12 dengan menggunakan *Wireshark* dimana titik-titik menunjukan paket data yang dikirimkan



Gambar 4. 12 Grafik *Throughput* pada *Google Maps*

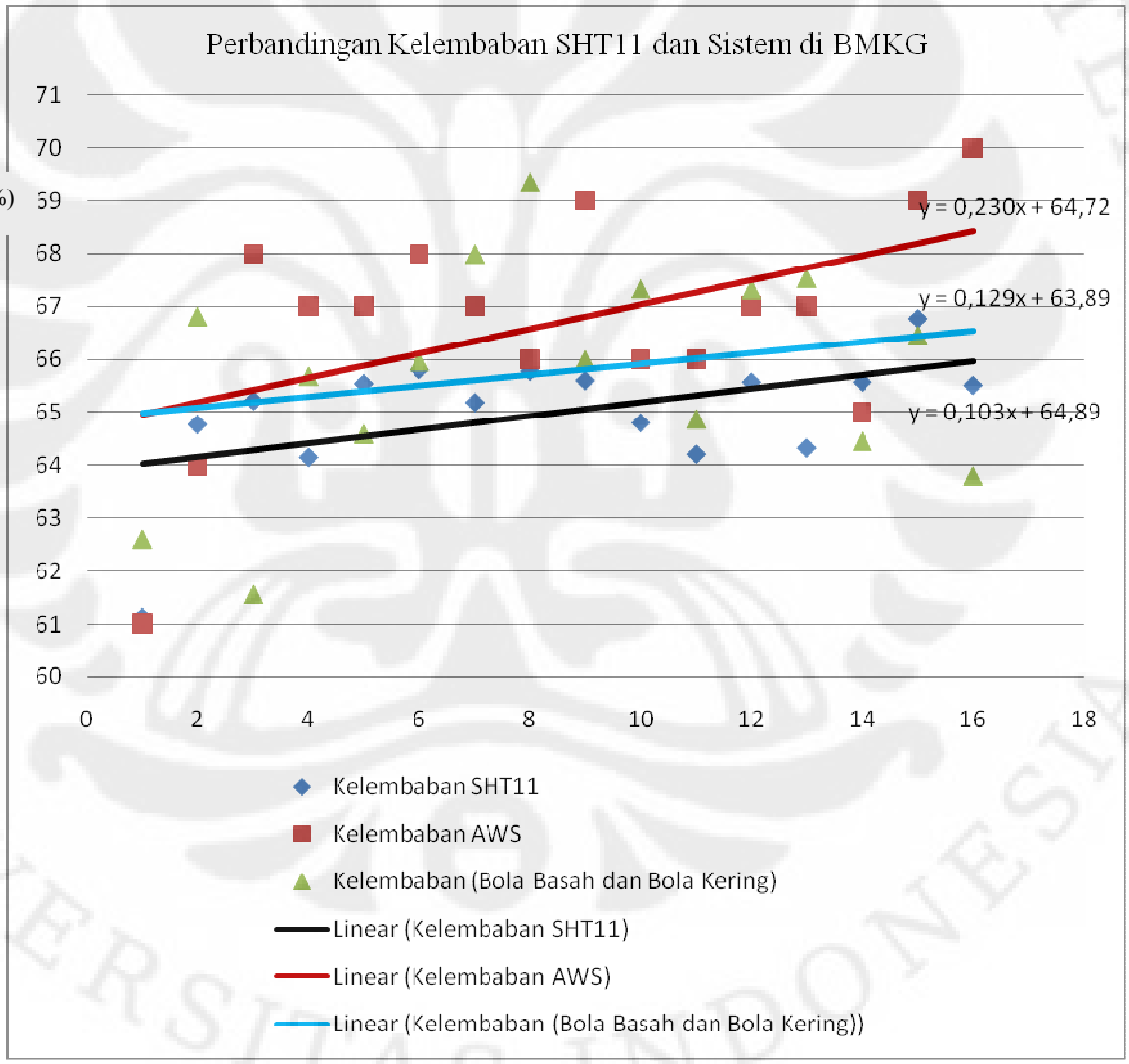
4.3 Uji Perbandingan

Uji perbandingan dimaksudkan agar terdapat perbandingan antara suhu dan kelembaban yang didapatkan dari sensor SHT11 dengan peralatan analog untuk mengukur suhu dan kelembaban udara di Badan Meteorologi dan Geofisika DKI Jakarta. Uji coba yang digunakan adalah dengan memvariasikan waktu dengan selang waktu pengambilan data adalah 10 menit di siang hari. Pengambilan datanya dilakukan mulai pukul 10.20 sampai 12.50 WIB Dari uji coba pengambilan data didapatkan hasil sebagai berikut :



Gambar 4.13 Grafik Perbandingan Suhu SHT11, Suhu AWS, dan Termometer Bola Basah-Bola Kering

Berdasarkan perbandingan yang dilakukan di Taman Alat, Stasiun Meteorologi, Badan Meteorologi, Klimatologi dan Geofisika (BMKG) Kemayoran Jakarta Pusat menghasilkan nilai presentase varian dari Suhu Sensor SHT11 dengan Sistem Elektronik AWS di BMKG perbedaan berkisar 3,15 ^{o}C terhadap sistem AWS (sistem elektronik pengukuran suhu dan kelembaban di BMG) dan 2,21 ^{o}C terhadap sistem analog (bola basah dan bola kering). Perbedaan ini cukup signifikan dalam pengukuran karena nilai perbedaannya mencapai 3 derajat celcius.



Gambar 4.14 Grafik Perbandingan Kelembaban SHT11, Suhu AWS, dan Termometer Bola Basah-Bola Kering

Sedangkan untuk data kelembaban, didapatkan perbedaan pengukuran adalah 2,2 % terhadap sistem AWS dan 2,0% terhadap sistem analog (bola basah dan bola kering) . Dari pengukuran yang telah dilakukan maka dapat disimpulkan bahwa sensor SHT11 yang digunakan masih belum siap untuk digunakan dilapangan karena besarnya kesalahan yang terjadi.

4.4 Uji Lapangan

Uji coba lapangan dilakukan di sekitar kampus Universitas Indonesia, Depok. Uji coba lapangan ini dimaksudkan mengetahui ketepatan lokasi dan pembacaan sensor dengan acuan bahwa GPS dapat 'memandang' langit tanpa penghalang. Pengujian ini merupakan pengujian *real time* dengan mengambil lokasi Universitas Indonesia dengan menggunakan kendaraan yang dibawa serta berkeliling. Dari percobaan terlihat bahwa sistem cukup lama untuk menyesuaikan ke posisi yang terbaru. Hal ini terjadi karena koneksi internet yang rendah dan juga rimbun pohon di UI yang membatasi akses terhadap GPS sehingga GPS akan menunjukan posisi yang sebelumnya. Pada *Google* Aplikasi akan lebih mudah menyesuaikan lokasinya karena yang berpengaruh hanya masalah akses GPS terhadap satelit, sedangkan peta lokasi telah tersimpan setelah aplikasi pertama kali dibuka. Pada aplikasi *Google Earth* yang berlisensi, peta pada saat aplikasi pertama kali dijalankan akan berjalan tanpa terus-terusan mengambil peta karena petanya sudah disimpan dalam aplikasi *Google Earth*.

Validasi per lokasi dan pembacaan sensor terangkum dalam tabel 4.1 di bawah ini :

Tabel 4. 1 Validasi per lokasi di Kampus UI Depok terhadap *Google Earth*

No	Lokasi	Lokasi Pada Peta	Pembacaan Sensor
1	Gedung Departemen Teknik Elektro	OK	OK
2	Halte Bus Kampus FT	OK	OK
3	Stadion UI	OK	OK
4	Halte Bus Kampus Stadion	OK	OK
5	Gymanasium UI	OK	OK
6	Halte Bus Kampus PNJ	OK	OK

7	Halte Bus Kampus FMIPA	OK	OK
8	Laboratorium ParangTopo	OK	OK
9	Halte Bus Kampus FKM	OK	OK
10	Gang Senggol	OK	OK
11	Halte Bus Kampus Balairung	OK	OK
12	Gerai ATM Balairung	OK	OK
13	Balairung	OK	OK
14	Rotunda	OK	OK
15	Rektorat	OK	OK
16	Menara Air	OK	OK
17	Perpustakaan Pusat	OK	OK
18	Halte Bus Kampus FIB	OK	OK
19	Teater Daun Cengkeh FIB	OK	OK
20	Pertigaan FASILKOM	OK	OK
21	Perpustakaan Pusat Baru	OK	OK
22	Masjid UI	OK	OK
23	Halte Bus Kampus MUI	OK	OK
24	Halte Bus Kampus FH	OK	OK
25	Tempat Parkir FT UI	OK	OK
26	Halte Bus Kampus FPsi	OK	OK
27	Halte Bus Kampus FISIP	OK	OK
28	Halte Bus Kampus PSJ	OK	OK
29	Halte Bus Kampus FE	OK	OK
30	Dekanat FT UI	OK	OK

Dari hasil pengujian per lokasi, tingkat keberhasilan sistem ini terhadap kondisi sebenarnya dilapangan adalah

$$\frac{30}{30} \times 100\% = 100\%$$

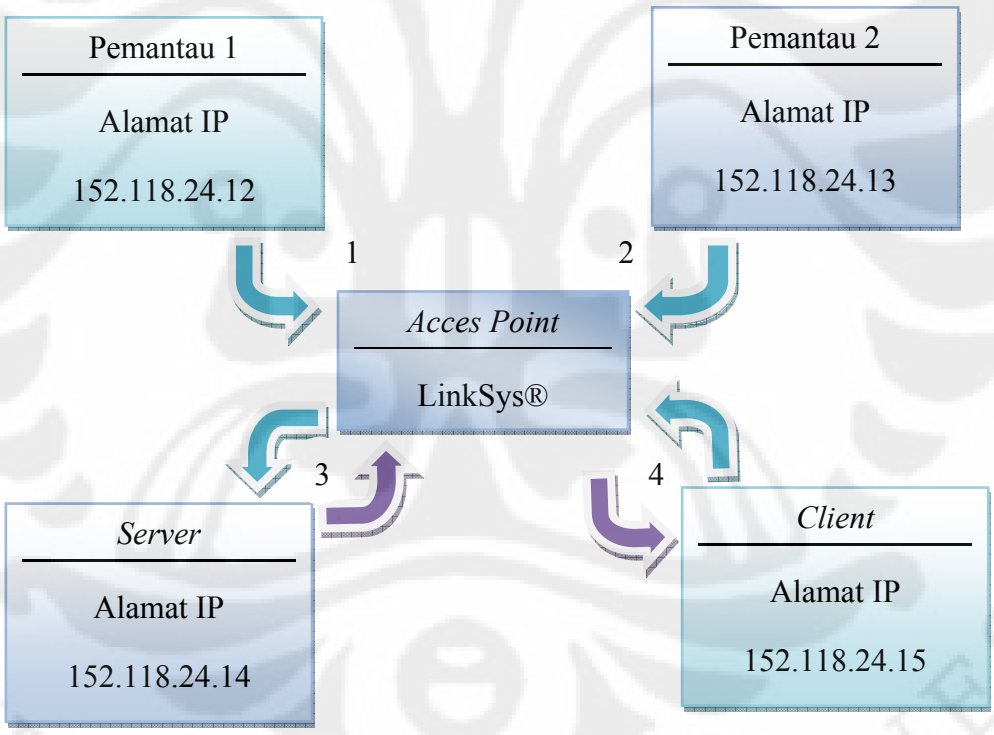
Sedangkan keberhasilan pembacaan sensor suhu dan kelembaban per lokasi adalah

$$\frac{30}{30} \times 100\% = 100\%$$

Sehingga dari pengujian ini sub sistem ini telah berjalan dengan baik karena tingkat keberhasilan pembacaan lokasi sebenarnya dan lokasi yang di tunjukan oleh GPS adalah sama.

4.5 Aplikasi Multi Sensor

Aplikasi multi sensor ini di susun dengan menggunakan *acces point* Linksys® dengan mengkonfigurasikan 2 laptop sebagai pengolah data GPS (representasi cuaca dalam keadaan yang sebenarnya), *server mysql,* dan *client.* Konsep percobaan aksesnya dapat dilihat pada gambar di bawah ini



Gambar 4. 15 Simulasi Percobaan aplikasi Multi sensor

Keterangan masing-masing proses pada gambar di atas adalah sebagai berikut :

1. Pemantau Cuaca 1 berisi sensor kelembaban dan suhu serta GPS dengan menggunakan alamat IP 152.118.24.12. Setiap ada perubahan keadaan maka dalam interval waktu tertentu (dalam percobaan ini setiap 1 detik)

maka terjadi perubahan data di *server* MySQL dengan koneksi sebelumnya melalui acces point sebagai representasi dari satelit.

2. Proses ini sama dengan proses no 1
3. Data pada *server* MySQL berubah sejalan dengan perubahan data sensor. Di *server* ini juga terdapat *file web*site *Google Maps*, pemroses KML, dan *networklink* yang nantinya akan di akses oleh komputer klien.
4. Komputer klien secara teratur mempebaharui data yang ditampilkan dari *server* MySQL melalui *acces point* yang ada. Data pada aplikasi *Google Maps* di perbaharui melalui fungsi *refresh javascript* dan *networklink* pada *Google Earth*.

Gambar 4. 16 Percobaan pada aplikasi multi sensor yang dilihat pada sisi *client*

Gambar 4.16 menggambarkan aplikasi *Google Earth* dengan banyak sumber dengan memanfaatkan wi-fi 802.11g. Sistem ini dapat berjalan baik dan dapat berkomunikasi dengan *server*. Pergerakan yang terlihat sangat kecil karena jangkauan dari akses jaringan nirkabel yang terbatas juga. Secara umum percobaan ini dapat berjalan dengan baik. Laptop 1 dan 2 menggunakan data dari

GPS, sedangkan laptop 3 menggunakan data *dummy* agar dapat dibandingkan dengan lokasi yang ada. Percobaan ini dilakukan di Kukusan Kelurahan.

Perbandingan dari Uji Coba yang dilakukan terhadap *Google Earth* dan *Google Maps* disajikan dalam tabel di bawah ini :

Tabel 4.2 Perbanding Kinerja Pemantauan Cuaca Melalui *Google Earth* dan *Google Maps*

No	Kinerja	*Google Earth*	*Google Maps*
1	Kinerja Pembaharuan Data	Pembaharuan data menggunakan mekanisme *Networklink* yang ada pada *Google Earth*	Pembaharuan data menggunakan elemen pembaharuan otomotasi dengan menggunakan Javascript
2	Interaktif antarmuka	Interaktif dan pembaharuan data tidak mempengaruhi keseluruhan peta	Cenderung tidak interaktif dan autorefresh dari *web* cukup mengganggu antarmuka
3	Sumber daya Internet	Lebih hemat sumber daya internet (*bandwith*) karena hanya membutuhkan banyak sumber daya pada inisialisasi awal	Membutuhkan sumber daya internet pada setiap pembaharuan data
4	Aplikasi *MultiClient*	Memungkinkan	Memungkinkan

BAB 5

KESIMPULAN

Berdasarkan Rancangan sistem dan hasil uji coba yang dilakukan di kampus Universitas Indonesia Depok serta uji kalibrasi di Stasiun BMKG Kemayoran Jakarta Pusat didapatkan kesimpulan sebagai berikut :

1. Rancang bangun sistem pemantauan cuaca dan parameter cuaca laut telah berhasil disimulasikan dengan mengambil lokasi di Kampus Universitas Indonesia Depok.
2. Sistem ini dapat berjalan dengan baik dan memberikan nilai pemantauan secara *real time* sesuai dengan posisi yang ditunjukan oleh GPS dan nilai yang diberikan oleh sensor.
3. Dari hasil uji validasi sensor, kesalahan pembacaan sensor SHT11 berkisar antara 2-3 ^{o}C untuk suhu dan 2-5% untuk kelembaban mengindikasikan bahwa sensor yang digunakan harus dikoreksi untuk mendapatkan pembacaan yang tepat.
4. Berdasarkan dari grafik TCP IO *Wireshark*, Grafik *Troughput Wireshark* serta uji aplikasi dengan Firebird menyimpulkan bahwa aplikasi *Google Earth* lebih sedikit membutuhkan sumber daya internet berdasarkan analisis waktu muat, *Troughput* , dan paket data yang dikirimkan sehingga cocok digunakan untuk aplikasi pemantauan cuaca.

DAFTAR REFERENSI

[1] ESRI (*Environmental Systems Research Institute, Inc.*).(n.d). 13 Desember 2009. http://www.esri.org.

[2] Mukhtar, A.Pi, M,Si. (2008).Pengaturan Penggunaan Sistem pemantauan Cuaca Perikanan (*Vessel Monitoring System*) Departemen Kelautan dan Perikanan,Kendari.

[3] *Vessel Monitoring System* (VMS).(2009). Majalah Biskom.

[4] *DKP Tetap Laksanakan Program VMS Bagi Cuaca Ikan.(*n.d). 16 November 2009.http://www. kapanlagi.com.

[5] *Global Positioning System*(2009).2 Desember 2009. http;//www.wikipedia.org

[6] Dulbahri.(1999). Sistem Informasi Geografi, Diktat Kuliah, PUSPICS Fakultas Geografi UGM, Yogyakarta, hal. 1-44

[7] Anisah Aini (2007). *Sistem Informasi Geografis dan Aplikasinya*. 11 Desember 2009. http://p3m.amikom.ac.id/.pdf.

[8] *Ibid*

[9] *Ibid*

[10] Botts, Mike; Alex Robin (Oct. 2007). *Bringing the Sensor Web Together*. Geosciences. pp. 46-53. http://www.brgm.fr/dcenews*File*?ID=473.

[11] Delin, Kevin; Shannon Jackson (2000). *Sensor Web for In Situ Exploration of Gaseous Biosignatures*. IEEE Aerospace Conference. http://www.sensorwaresystems.com/historical/resources/sensor*web*-concept.pdf.

[12] Delin, Kevin (2005). *Sensor Webs in the Wild*. Wireless Sensor Networks: A Systems Perspective Artech House. http://www.sensorwaresystems.com/historical/resources/DelinSensor*Webs*InTheWildChapter2005.pdf.

[13] *Datasheet SHT11*.(n.d).24 Oktober 2009.http://www.sensirion.com.

[14] *Global Positioning System*. (2009). 2 Desember 2009. http;//www.wikipedia.org

[15] *Ibid*

[16] XAMPP.(2009). XAMPP [Piranti Lunak Komputer]. http://www.apachefriends.org

[17] Bordland Corporation (2004). Delphi® 7.0©[Piranti Lunak Komputer]

[18] Tim IE(2004). *Buku Panduan Penggunaan Produk Innovative Electronics*. 4 November 2009.http://www.innovativeelectronics.com.

[19] *Ibid*

[20] *Google Earth*.(2009). 12 Oktober 2009. http://id.wikipedia.org/wiki/*Google*_Earth.

[21]*Google* Inc(2009).*Google Earth* [Piranti Lunak Komputer]

[22]Tim *Google* (2009). *Networklink*. 2 Desember 2009. http://code.*Google*.com/apis/kml/documentation/kml_tut.html

[23]*Ibid*

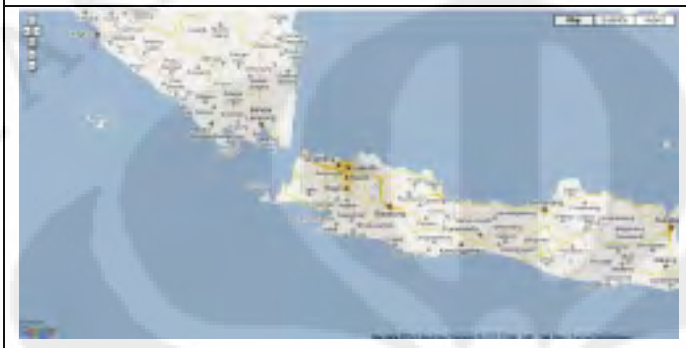
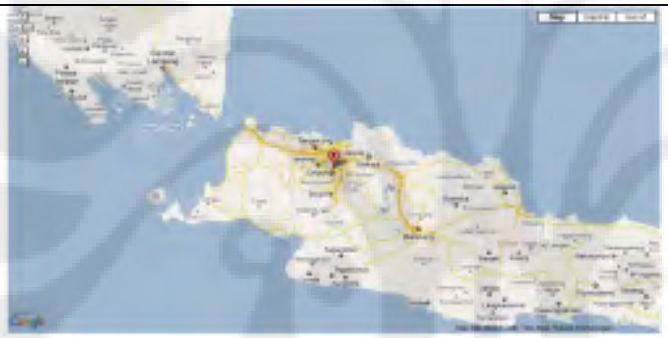
DAFTAR PUSTAKA

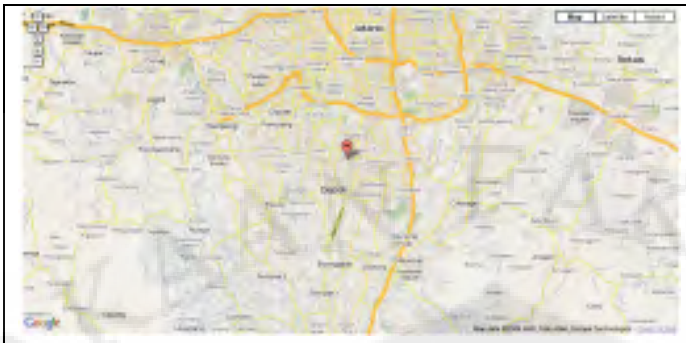
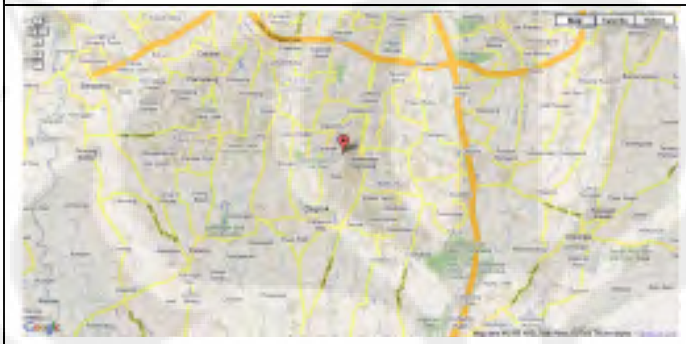
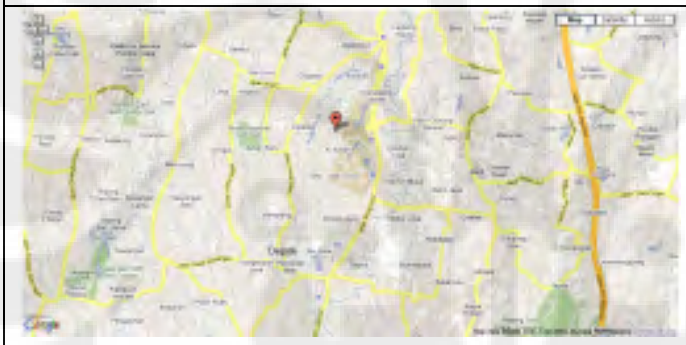
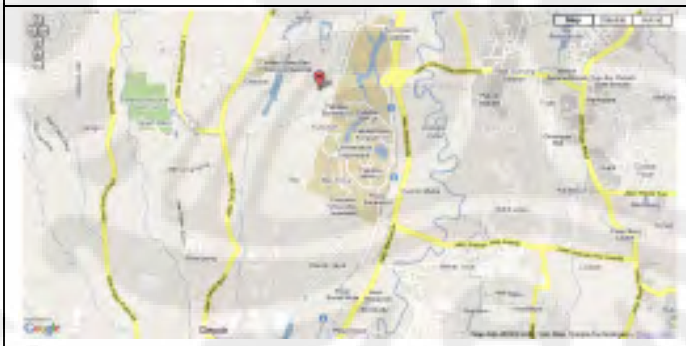
Sorribas J, dkk.(2009). *Real-Time Fleet Ship Monitoring System using Satellite Broadband Communications and Google Earth.2009 First International Conference on Advances in Satellite and Space Communications*. Barcelona, Spanyol.

Sorribas J, dkk.(2009). *Satellite Communication Systems Onboard Spanish Oceanographic Vessels Performance and optimization analysis.2009 First International Conference on Advances in Satellite and Space Communications*. Barcelona, Spanyol.

DAFTAR LAMPIRAN
LAMPIRAN 1

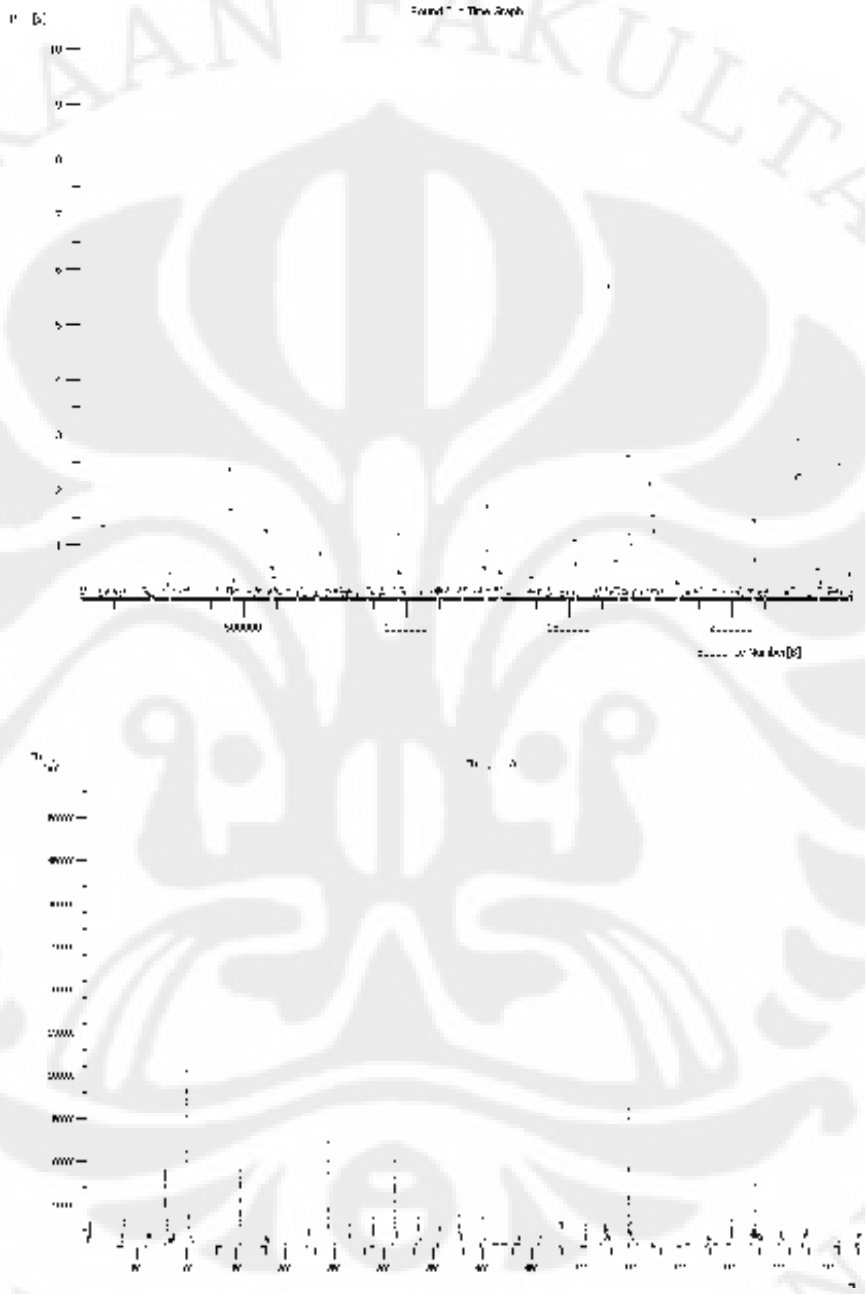
REPRESENTASI GAMBAR KOEFISIEN PERBESARAN

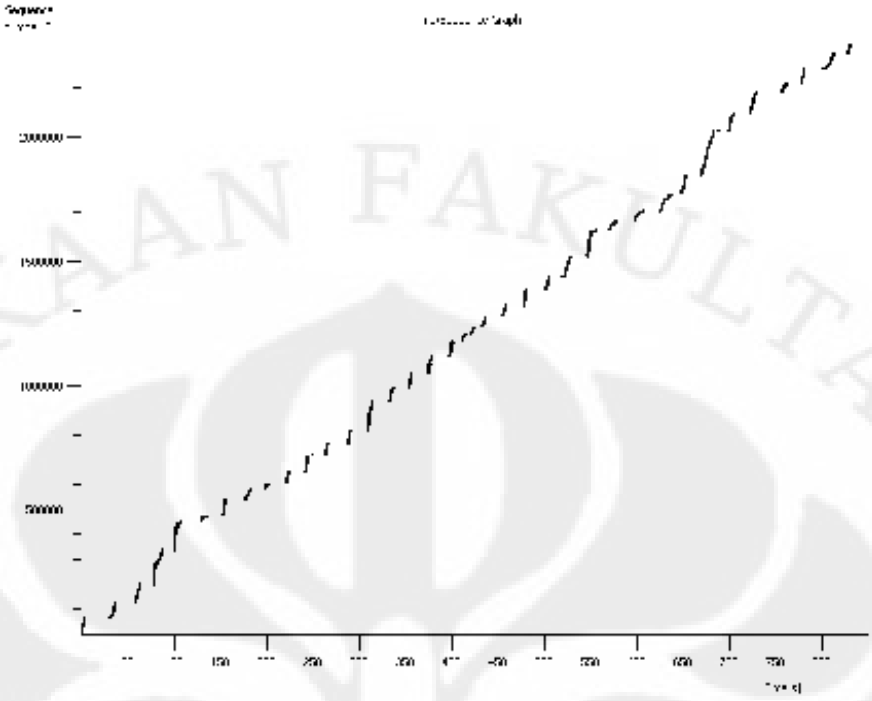
Objek Gambar	Koefisien Perbesaran
	map.setCenter(new GLatLng(latitude, longitude), 7);
	map.setCenter(new GLatLng(latitude, longitude), 8);
	map.setCenter(new GLatLng(latitude, longitude), 9);
	map.setCenter(new GLatLng(latitude, longitude), 10);

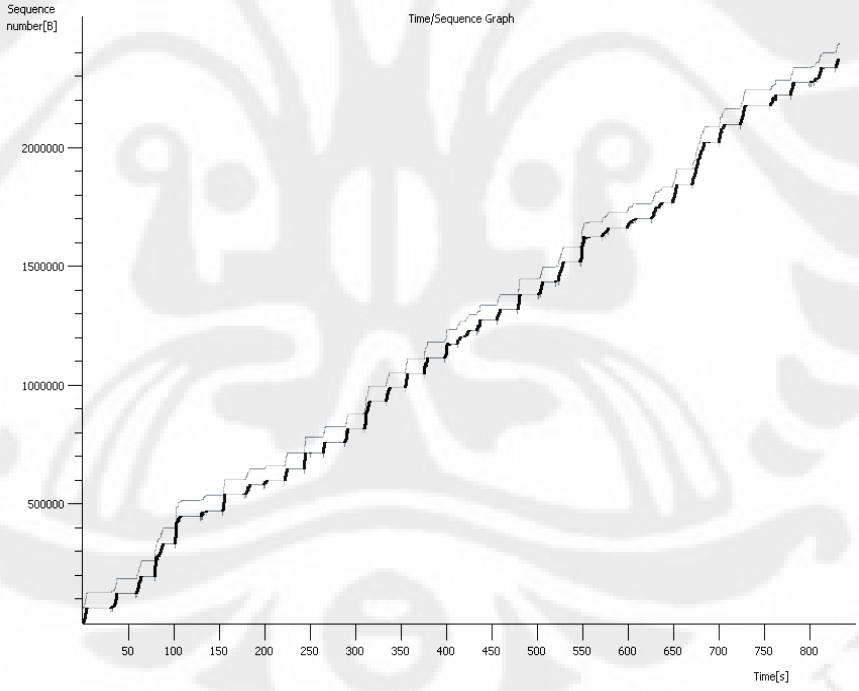
	map.setCenter(new GLatLng(latitude, longitude), 11);
	map.setCenter(new GLatLng(latitude, longitude), 12);
	map.setCenter(new GLatLng(latitude, longitude), 13);
	map.setCenter(new GLatLng(latitude, longitude), 14);

LAMPIRAN 2

TCP STREAM GRAPH *WIRESHARK* UNTUK APLIKASI *GOOGLE MAPS*

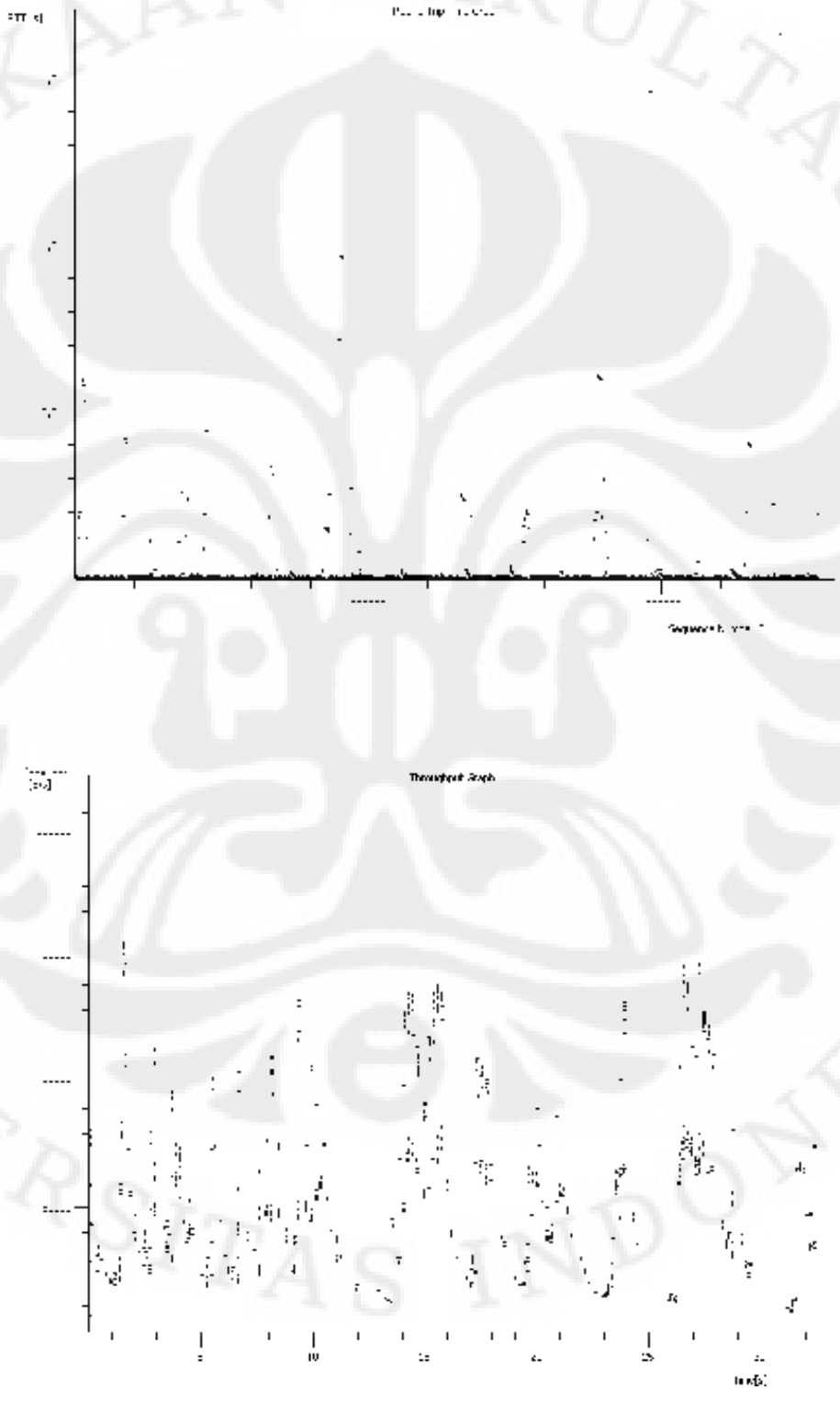


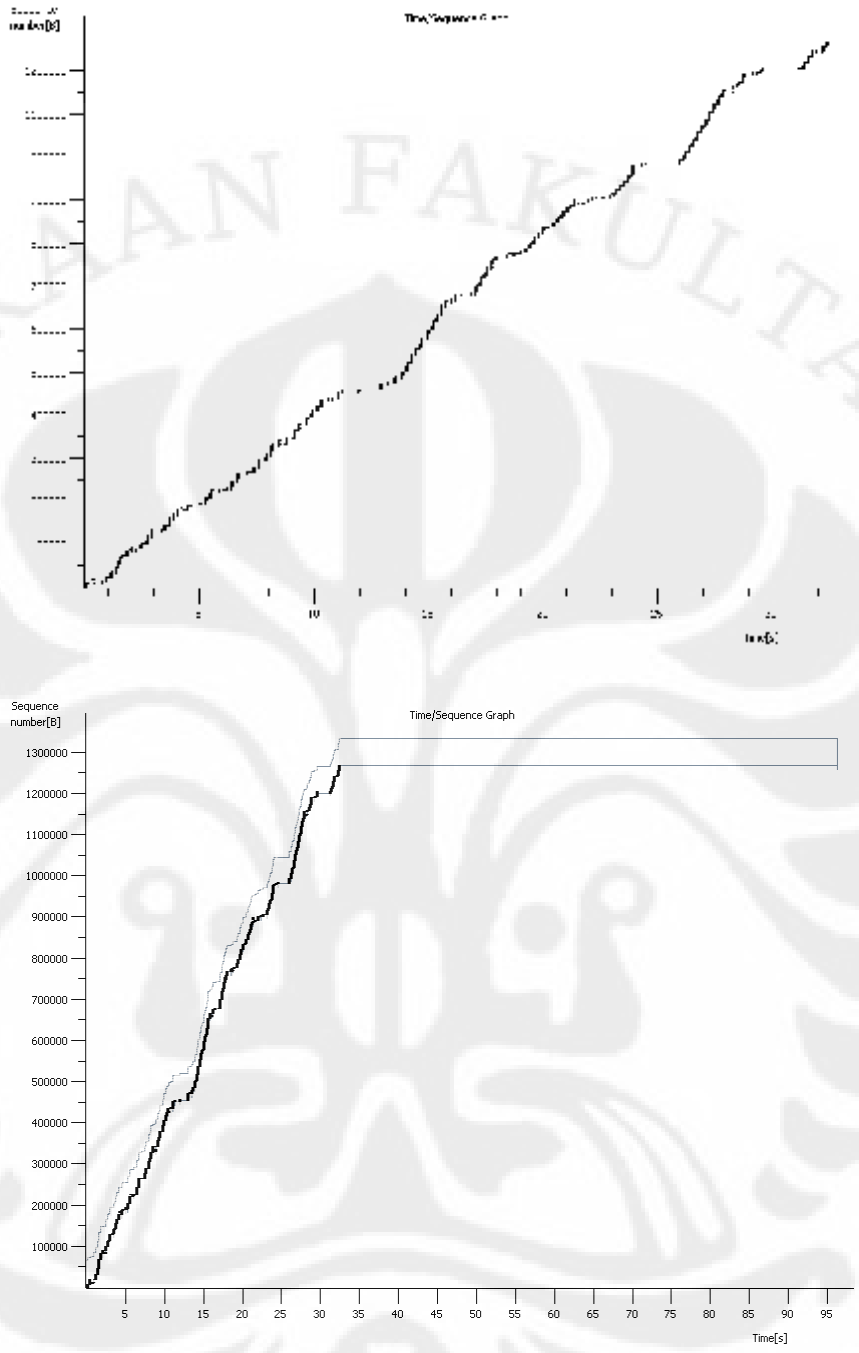



Sequence number[B]
Time/Sequence Graph
2000000
1500000
1000000
500000
50
100
150
200
250
300
350
400
450
500
550
600
650
700
750
800
Time[s]

LAMPIRAN 3

TCP STREAM GRAPH *WIRESHARK* UNTUK APLIKASI *GOOGLE EARTH*




Sequence number[B]
Time/Sequence Graph
1300000
1200000
1100000
1000000
900000
800000
700000
600000
500000
400000
300000
200000
100000
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
Time[s]

LAMPIRAN 4

PENGOLAHAN DATA SENSOR SUHU

No	Waktu	Pengukuran SHT11		Pengukuran Bola Basah dan Bola Kering				AWS (Sistem		Selisih Error Kuadrat			
										BB dan BK		AWS	
		Suhu(°C)	Kelembaban(%)	Suhu Bola Kering(°C)	Suhu Bola Basah(°C)	Suhu(°C)	Kelembaban(%)	Suhu(°C)	Kelembaban(%)	Suhu	Kelembaba	Suhu	Kelembaba
1	10:20:00	34,00	61,1309	32,2	26,2	32,2	62,6	32,2	61	3,24	2,1582548	3,24000	0,0171348
2	10:30:00	34,44	64,7711	31,4	26,2	31,4	66,81	31,6	64	9,24	4,1571132	8,06560	0,5945952
3	10:40:00	34,24	65,2129	32,4	26,2	32,4	61,55	30,9	68	3,39	13,416836	11,15560	7,7679264
4	10:50:00	34,57	64,1504	32,1	26,9	32,1	65,68	31,4	67	6,10	2,3396762	10,04890	8,1202202
5	11:00:00	34,45	65,5361	32,4	26,9	32,4	64,59	31,4	67	4,20	0,8951052	9,30250	2,1430032
6	11:10:00	34,53	65,8	32,6	27,2	32,6	65,96	31,7	68	3,72	0,0256	8,00890	4,84
7	11:20:00	34,75	65,1835	32,8	27,3	32,8	67,99	31,6	67	3,80	7,8764423	9,92250	3,2996723
8	11:30:00	34,93	65,7707	32,6	27,7	32,6	69,35	31,7	66	5,43	12,811388	10,43290	0,0525785
9	11:40:00	35,09	65,5948	32,8	27,4	32,8	65,99	31,9	69	5,24	0,156183	10,17610	11,595387
10	11:50:00	35,53	64,8006	32,8	27,6	32,8	67,33	32,1	66	7,45	6,3978644	11,76490	1,4385604
11	12:00:00	35,56	64,2096	32,8	27,2	32,8	64,87	31,7	66	7,62	0,4361282	14,89960	3,2055322
12	12:10:00	35,18	65,5654	32,8	27,6	32,8	67,32	31,7	67	5,66	3,0786212	12,11040	2,0580772
13	12:20:00	35,07	64,328	32,6	27,4	32,6	67,53	31,9	67	6,10	10,252804	10,04890	7,139584
14	12:30:00	34,64	65,5654	33	27,2	33	64,45	31,6	65	2,69	1,2441172	9,24160	0,3196772
15	12:40:00	34,57	66,7639	33	27,6	33	66,45	31,1	69	2,46	0,0985332	12,04090	5,0001432
16	12:50:00	34,64	65,5067	33,1	27,6	33,1	63,8	31,6	70	2,37	2,9128249	9,24160	20,189745

LAMPIRAN 4 (LANJUTAN)

PENGOLAHAN DATA SENSOR SUHU

Keterangan	Olah Data Suhu BMG AWS dan SHT11	Olah Data Kelembaban BMG AWS dan SHT11	Olah data Suhu BMG BB BK dan SHT11	olah data kelembaban BMG BB BK dan SHT11
Varian BMG	0,108958333	4,629166667	0,179833333	4,199862917
Varian SHT11	0,18957625	1,521405351	0,18957625	1,521405351
Varian BMG- VarianSHT11	-0,080617917	3,107761315	-0,009742917	2,678457565
Hasil Bagi	-0,73989675	0,671343578	-0,054177479	0,637748807
Kurang	0,73989675	0,671343578	0,054177479	0,637748807
Presentase Varian	73,98967495	67,13435785	5,417747915	63,77488072
Jumlah Error Kuadrat	159,70090	77,78183652	78,7329	68,25749252
Jumlah Data	16	16	16	16
RMS	3,159320536	2,204850286	2,218289037	2,065452319

LAMPIRAN 5

UJI MUAT APLIKASI *GOOGLE EARTH*

Uji Waktu Muat Aplikasi		
No,	Waktu Muat (detik)	Kecepatan Unduh (kbps)
1	2,85	249
2	2,38	21
3	4,43	470
4	1,63	157
5	1,48	180
6	1,25	373
7	2,37	399
8	3,57	377
9	1,36	756
10	1,45	91
11	3,17	28
12	2,07	165
13	2,19	466
14	1,5	21
15	3,63	21
16	3,77	156
17	2,12	20
18	1,61	316
19	0,99	277

LAMPIRAN 6

UJI MUAT APLIKASI *GOOGLE MAPS* DIVARIASIKAN TERHADAP PERBESARAN

Percobaan	Waktu Muat (detik) pada tingkat pembesaran							
n	7	8	9	10	11	12	13	14
1	2,18	2,51	5,01	4,11	5,44	9,77	14,76	7,27
2	1,94	2,56	5,35	13,44	5,64	6,95	4,70	6,65
3	2,00	3,03	4,41	10,39	2,91	4,55	6,98	9,58
4	1,98	2,49	4,24	4,89	3,74	5,48	9,81	7,86
5	1,86	2,73	4,70	3,38	4,75	5,69	6,89	8,48
6	1,90	2,72	4,24	3,95	2,85	7,54	4,65	7,41
7	2,07	3,39	4,24	4,66	3,47	7,16	8,01	7,01
8	2,12	3,13	4,48	3,90	2,80	7,23	7,25	9,29
9	1,93	2,82	4,42	3,27	4,55	8,67	16,30	6,92
10	1,97	2,70	3,69	4,58	4,41	6,39	4,94	7,71
11	2,18	2,67	4,46	3,92	4,51	7,87	6,85	8,22
12	2,03	2,86	5,65	4,54	2,82	4,20	7,54	4,70
13	2,54	3,01	2,58	4,74	5,31	4,87	11,81	6,99
14	2,47	2,22	3,92	4,32	3,78	4,30	7,65	6,71
15	2,34	2,63	5,10	3,77	6,16	4,00	7,09	6,86
16	2,56	3,26	4,69	3,36	5,10	6,52	7,61	7,77
17	2,36	3,18	4,37	4,54	6,55	11,17	12,63	8,87
18	1,98	2,72	4,02	7,07	4,95	5,23	13,01	13,51
19	2,17	3,01	5,18	5,14	5,07	6,88	8,36	8,52
20	1,96	2,92	5,21	4,40	3,61	5,60	8,20	7,79
21	2,17	2,76	4,08	4,45	4,61	5,63	8,20	7,32
22	2,30	3,14	4,58	3,80	4,87	8,01	4,93	6,86
23	2,35	3,10	4,12	4,74	6,24	4,85	5,18	8,20
24	2,78	3,42	4,72	4,63	4,64	5,87	7,55	6,81
25	2,37	3,13	5,01	5,61	5,18	7,11	8,77	8,17
26	5,65	4,34	3,95	3,71	5,12	8,15	10,98	8,56
27	2,21	3,01	5,13	3,22	7,91	7,09	5,44	8,30
28	2,16	2,55	4,99	4,54	6,29	5,67	9,72	8,78
29	2,19	4,70	4,70	4,17	5,02	5,67	8,74	8,21
30	2,00	3,81	4,46	3,54	5,80	5,43	5,35	8,48
Rata-Rata	2,291	3,017	4,5233	4,826	4,80333	6,451667	8,33	7,927

LAMPIRAN 7

SOURCE CODE APLIKASI MIKROKONTROLER

(merupakan modifikasi dari source code yang dibuat oleh tim Innovative Electronic. http://www.innovativeelectronics.com)

```
/* Chip type         : ATmega8535
  Program type       : Application
  Clock frequency    : 4,000000 MHz
  Memory model       : Small
  External SRAM size  : 0
  Data Stack size    : 128       */


#include <mega8535.h>
#include <delay.h>
#include <stdio.h>   // Standard Input/Output functions

sfrb PORTA=0x1b;
sfrb PINA=0x19;
#define SDAOut PORTA.0
#define SDAIn PINA.0
#define SCLK PORTA.1

// Declare your global variables here
unsigned char DataTRX,TimeOut,AckBit;  //Ackbit : '0' (ACK), '1' (NOACK)
unsigned int DataTempSHT,DataRHSHT,DataRead;

void Transmitte (void)
{
unsigned char UCSRATemp;
Wait4ClearedUDRE:
   UCSRATemp = UCSRA;
   UCSRATemp &= 0b00100000;
   if(UCSRATemp == 0b00100000)
     {
     delay_ms(100);
     UDR = DataTRX;    //Send a character in var dataTR
     }
   else goto Wait4ClearedUDRE;
}

/*Membuat kondisi "start" ke SHT11 ("Transmission Start" sequence)
      _____     _____
 ShtData     |_____|

         ___   ___
 ShtClock ___|   |___|   |___
 Clock       1     2      */
void StartSignal (void)
{
unsigned char DDRATemp;
```

```
   DDRATemp = DDRA;
   DDRA |= 0x01;      // PortA.0 sbg Output
   SDAOut = 1;
   SCLK = 0;
   SCLK = 1;         //Clock pertama
   SDAOut = 0;
   SCLK = 0;
   SCLK = 1;         //Clock kedua
   SDAOut = 1;
   SCLK = 0;         //Pin Clock = '0'
   DDRA = DDRATemp;
}

//Reset komunikasi: 9 clock cyle dengan ShtData '1', lalu kondisi start
void ResetSHT (void)
{
unsigned char i,DDRATemp;
   DDRATemp = DDRA;
   DDRA |= 0x01;
   SDAOut = 1;
   SCLK = 0;
   for (i=0; i<=8; i++)
    {
      SCLK = 1;    //Kirim Data (ShtClock rising edge), 9 kali
      SCLK = 0;
    }
   StartSignal();       //Transmission Start
   DDRA = DDRATemp;
}

//Tunggu sampai SHT11 selesai melakukan pengukuran (pin Data = '0')
//Timeout pengukuran sekitar 1/4 detik (TimeOut = '0' --> measure OK)
void SHTWait (void)
{
   unsigned char i,DDRATemp;
   DDRATemp = DDRA;
   DDRA |= 0x01;
   SDAOut=1;         //Pin ShtData sebagai input
   DDRA &= 0xFE;
   for (i=0; i<250; i++)
    {
      TimeOut=SDAIn;       //Jika pin ShtData = '0' --> pengukuran selesai
      if (TimeOut==0) goto ExitSHT_Wait;
      delay_ms(1);
    }
   ExitSHT_Wait:
   DDRA = DDRATemp;
}

// Transmit Data dan ambil bit Acknowledge
void SHTWriteByte (unsigned char data)
{
   unsigned char i,DDRATemp;
   DDRATemp = DDRA;
   DDRA |= 0x01;
```

```
  for (i=0; i<8; i++)
   {
    if ((data>>7)==1) SDAOut = 1;   //Kirim MSB first
    else SDAOut = 0;
    SCLK = 1;    //Kirim Data (ShtClock rising edge)
    SCLK = 0;
    data <<= 1; // geser data kekiri 1 bit
   }
  SDAOut = 1;        //Pin ShtData sebagai input
  SCLK = 1;
  DDRA &= 0xFE;
  AckBit = SDAIn;    //Ambil sinyal acknowledge
  SCLK = 0;
  DDRA = DDRATemp;
}

//Receive Data dan kirim bit "AckBit" ('0' untuk ACK atau '1' untuk NACK)
void SHTReadByte (void)
{
  unsigned char i,DDRATemp;
  DataRead = 0x00;
  DDRATemp = DDRA;
  DDRA |= 0x01;
  SDAOut = 1;        //Pin ShtData sebagai input
  DDRA &= 0xFE;
  for (i=0; i<8; i++)
   {
    DataRead<<=1;
    SCLK = 1;
    DataRead |= SDAIn;  //Ambil Data (MSB first)
    SCLK = 0;
   }
  DDRA |= 0x01;
  if (AckBit==1) SDAOut = 1;  //Kirim Noacknowledge
  else SDAOut = 0;          //Kirim Acknowledge
  SCLK = 1;
  SCLK = 0;
  SDAOut = 1;        //Pin ShtData sebagai input
  DDRA = DDRATemp;
}

// Pembacaan Temperature dari SHt11
void SHTReadTemp (void)
{
  StartSignal();
  SHTWriteByte(0x03);   //Command Measure Temperature
  if (AckBit==0)
   {
    SHTWait();       //Tunggu sampai pengukuran selesai
    if (TimeOut==0)
     {
      AckBit=0;              //Kirim ACK untuk menerima byte berikutnya
      SHTReadByte();          // Ambli Byte MSB
      DataTempSHT = DataRead;
      DataTempSHT <<= 8;
```

```
        AckBit=1;              //Kirim NACK untuk mengakhiri pengambilan data
        SHTReadByte();
        DataTempSHT |= DataRead;  //Ambil byte LSB

        DataRead = DataTempSHT;
        DataRead >>= 8;
        delay_ms(100);
        DataTRX = DataRead;
        Transmitte();

        DataRead = DataTempSHT;
        DataRead &= 0xFF;
        delay_ms(100);
        DataTRX = DataRead;
        Transmitte();
       }
    }
}

void SHTReadHumidity (void)
{
   StartSignal();
   SHTWriteByte(0x05);  //Command Measure Humidity
   if (AckBit==0)
    {
      SHTWait();
      if (TimeOut==0)
       {
        AckBit=0;
        SHTReadByte();
        DataRHSHT = DataRead;
        DataRHSHT <<= 8;
        AckBit=1;
        SHTReadByte();
        DataRHSHT |= DataRead;

        DataRead = DataRHSHT;
        DataRead >>= 8;
        DataTRX = DataRead;
        delay_ms(100);
        Transmitte();

        DataRead = DataRHSHT;
        DataRead &= 0xFF;
        DataTRX = DataRead;
        delay_ms(100);
        Transmitte();
       }
    }
}

void main(void)
{
// Declare your local variables here
unsigned char UCSRA_Temp,DataSerial;
```

```
// Hub DT-AVR dg Sensirion SHT11
// Port A --> A.1 = Output(SCLK), A.0 = Input/Output(SDA)
PORTA=0x00;
DDRA=0x02;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud rate: 9600 (Double Speed Mode)
UCSRA=0x02;
UCSRB=0x18;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x33;

delay_ms(1000);
ResetSHT();     //Connection Reset

while (1)
    {
    // Place your code here

     UCSRA_Temp = UCSRA;
     UCSRA_Temp &= 0x80;
     if (UCSRA_Temp == 0x80)
      {
      DataSerial = UDR;          // Klo gak dibaca loop-ing terus
      delay_ms(1000);
      if (DataSerial == 1)
        { SHTReadTemp(); ResetSHT(); }
      else if (DataSerial == 2)
        { SHTReadHumidity(); ResetSHT(); }
      else { ResetSHT(); }
      }
    };
}
```

LAMPIRAN 8

SOURCE CODE APLIKASI DELPHI

(merupakan modifikasi dari source code yang dibuat oleh tim Innovative Electronic untuk pembacaan SHT11 dan komponen TGPS untuk pembacaanGPS dengan penambahan komponen Database MySQL)

```
unit Main;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, GPS, CommDriver, ComCtrls, ExtCtrls, GPSSatData, SatDataBase,
  ShellAPI, Registry, DB, ADODB, CPort, XPMan, ZConnection,
  ZAbstractRODataset, ZAbstractDataset, ZDataset;
type
  TForm1 = class(TForm)
    GPS1: TGPS;
    Button1: TButton;
    Button2: TButton;
    lblStat: TLabel;
    dlgSave: TSaveDialog;
    lblStatus: TStaticText;
    Timer1: TTimer;
    CP1: TComPort;
    Timer2: TTimer;
    XPManifest1: TXPManifest;
    Label61: TLabel;
    waktusekarang: TTimer;
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    Label42: TLabel;
```

Label9: TLabel;

GroupBox1: TGroupBox;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

Label8: TLabel;

edGGA1: TEdit;

edGGA2: TEdit;

edGGA5: TEdit;

edGGA6: TEdit;

edGGA7: TEdit;

edGGA3: TEdit;

edGGA4: TEdit;

edGGA8: TEdit;

GroupBox3: TGroupBox;

Label14: TLabel;

Label15: TLabel;

Label16: TLabel;

Label17: TLabel;

edGLL3: TEdit;

edGLL1: TEdit;

edGLL2: TEdit;

edGLL4: TEdit;

GroupBox4: TGroupBox;

Label19: TLabel;

Label20: TLabel;

Label21: TLabel;

Label22: TLabel;

Label23: TLabel;

Label24: TLabel;

edRMC1: TEdit;

edRMC2: TEdit;

edRMC3: TEdit;

edRMC4: TEdit;

edRMC5: TEdit;

edRMC6: TEdit;

GroupBox5: TGroupBox;

Label25: TLabel;

Label26: TLabel;

Label27: TLabel;

Label28: TLabel;

Label29: TLabel;

Label30: TLabel;

Label31: TLabel;

Label32: TLabel;

Label33: TLabel;

Label34: TLabel;

edRMB3: TEdit;

edRMB2: TEdit;

edRMB1: TEdit;

edRMB6: TEdit;

edRMB5: TEdit;

edRMB4: TEdit;

edRMB9: TEdit;

edRMB8: TEdit;

edRMB7: TEdit;

edRMB10: TEdit;

GroupBox6: TGroupBox;

Label35: TLabel;

Label36: TLabel;

edUnh2: TEdit;

edUnh1: TEdit;

GPSSatData2: TGPSSatData;

GPSSatSignals2: TGPSSatSignals;

TabSheet2: TTabSheet;

lblAutoDetect: TLabel;

Label54: TLabel;

rgComPort: TRadioGroup;

rgComSpeed: TRadioGroup;

rgDatabits: TRadioGroup;

rgStopbits: TRadioGroup;

rgParity: TRadioGroup;

btnAutodetect: TButton;

cbDTR: TCheckBox;

cbRTS: TCheckBox;

btnStopAutodetect: TButton;

TabSheet5: TTabSheet;

Label55: TLabel;

Label56: TLabel;

derja: TLabel;

Label57: TLabel;

Label58: TLabel;

Label59: TLabel;

Label60: TLabel;

txtsuhu: TEdit;

txtkelembaban: TEdit;

```
btnconnect: TButton;
btnsetting: TButton;
edit1: TEdit;
dewpoint: TEdit;
cuaca: TEdit;
TabSheet3: TTabSheet;
ZConnection1: TZConnection;
ZQuery1: TZQuery;
GroupBox2: TGroupBox;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
Label13: TLabel;
server: TEdit;
user: TEdit;
pass: TEdit;
dbase: TEdit;
Button3: TButton;
GroupBox7: TGroupBox;
Button6: TButton;
GroupBox8: TGroupBox;
Label18: TLabel;
id: TEdit;
Button4: TButton;
Timer3: TTimer;
Button5: TButton;
procedure Button1Click(Sender: TObject);
procedure GPS1Position(Sender: TObject; FixTime: TDateTime; Latitude,
 Longitude: Double; Quality: TGPSQuality; SatCount: Integer; HDOP,
 Altitude, GeodidHeight: Double);
```

```
procedure Button2Click(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure rgComPortClick(Sender: TObject);
procedure rgComSpeedClick(Sender: TObject);
procedure GPS1GeoPosition(Sender: TObject; Valid: Boolean;
 FixTime: TDateTime; Latitude, Longitude: Double);
procedure GPS1MinGPSData(Sender: TObject; Warning: Boolean;
 FixDateTime: TDateTime; Latitude, Longitude, Speed, Course,
 MagVariation: Double);
procedure GPS1Unhandled(Sender: TObject; Talker, ID: String;
 Data: TStringList);
procedure Button3Click(Sender: TObject);
        procedure rgDatabitsClick(Sender: TObject);
procedure rgStopbitsClick(Sender: TObject);
procedure rgParityClick(Sender: TObject);
procedure GPS1NavigationInfo(Sender: TObject; Warning,
 Arrived: Boolean; CTE: Double; OriginWaypoint, DestWaypoint: String;
 DestLat, DestLon, RangeDest, BearingDest, Velocity: Double);
procedure btnAutodetectClick(Sender: TObject);
procedure cbDTRClick(Sender: TObject);
procedure btnStopAutodetectClick(Sender: TObject);
procedure GPS1Status(Sender: TObject; Status: TNMEAStatus);
procedure Timer1Timer(Sender: TObject);
procedure Button6Click(Sender: TObject);
procedure btnsettingClick(Sender: TObject);
procedure btnconnectClick(Sender: TObject);
procedure IntTim2(Sender: TObject); // baru
procedure IntSerialRX(Sender: TObject; Count: Integer);
procedure waktusekarangTimer(Sender: TObject);
procedure Timer3Timer(Sender: TObject);
```

```
  procedure Button4Click(Sender: TObject);
  procedure Button5Click(Sender: TObject); //baru
        private
  { Private declarations }
        public
  { Public declarations }
        end;
var
 Form1: TForm1;
 DataSerial: Integer;
 DataSerialTemp: Integer;
 FlagRX: Integer;
 DataTrans: Integer;
 DataSHT: Real;
 Temp: Real;
implementation
{$R *.DFM}
procedure TForm1.Button1Click(Sender: TObject);
begin
 Close;
end;
procedure TForm1.GPS1Position(Sender: TObject; FixTime: TDateTime;
 Latitude, Longitude: Double; Quality: TGPSQuality; SatCount: Integer;
 HDOP, Altitude, GeodidHeight: Double);
begin
 edGGA1.Text := TimeToStr(FixTime);
 edGGA2.Text := Format('%.4f',[Latitude]);
 edGGA3.Text := Format('%.4f',[-Longitude]);
 case Quality of
  gpsqNone:
```

```
    edGGA4.Text := 'None';
   gpsqNormal:
    edGGA4.Text := 'Normal';
   gpsqDifferential:
    edGGA4.Text := 'Diff';
  end;
  edGGA5.Text := Format('%.2f',[HDOP]);
  edGGA6.Text := Format('%.2f',[Altitude]);
  edGGA7.Text := Format('%.2f',[GeodidHeight]);
  edGGA8.Text := IntToStr(SatCount);
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  GPS1.Connected := True;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  rgComPort.ItemIndex := 0;
  rgComSpeed.ItemIndex := 5;
  rgDatabits.ItemIndex := 1;
  rgStopBits.ItemIndex := 0;
  rgParity.ItemIndex := 0;
end;
procedure TForm1.GPS1GeoPosition(Sender: TObject; Valid: Boolean;
  FixTime: TDateTime; Latitude, Longitude: Double);
begin
  edGLL1.Text := Format('%.4f',[Latitude]);
  edGLL2.Text := Format('%.4f',[Longitude]);
```

```
 edGLL3.Text := TimeToStr(FixTime);
 if Valid then
  edGLL4.Text := 'Yes'
 else
  edGLL4.Text := 'No';
end;
procedure TForm1.GPS1MinGPSData(Sender: TObject; Warning: Boolean;
 FixDateTime: TDateTime; Latitude, Longitude, Speed, Course,
 MagVariation: Double);
begin
 edRMC1.Text := DateTimeToStr(FixDateTime);
 edRMC2.Text := Format('%.4f',[Latitude]);
 edRMC3.Text := Format('%.4f',[Longitude]);
 if Warning then
  edRMC4.Text := 'Yes'
 else
  edRMC4.Text := 'No';
 edRMC5.Text := Format('%.2f',[Speed]);
 edRMC6.Text := Format('%.2f',[Course]);
end;

procedure TForm1.GPS1Unhandled(Sender: TObject; Talker, ID: String; Data: TStringList);
begin
 edUnh1.Text := Talker;
 edUnh2.Text := ID;
end;
procedure TForm1.Button3Click(Sender: TObject);
begin
zconnection1.HostName := server.Text;
zconnection1.User := user.Text;
```

```
zconnection1.Password := pass.Text;
zconnection1.Database := dbase.Text;
zconnection1.Connected := True;
end;
procedure TForm1.rgComPortClick(Sender: TObject);
begin
 case rgComPort.ItemIndex of
   0: GPS1.Port := cnCOM1;
   1: GPS1.Port := cnCOM2;
   2: GPS1.Port := cnCOM3;
   3: GPS1.Port := cnCOM4;
 end;
end;
procedure TForm1.rgComSpeedClick(Sender: TObject);
begin
 case rgComSpeed.ItemIndex of
   0: GPS1.PortBaud := 300;
   1: GPS1.PortBaud := 600;
   2: GPS1.PortBaud := 1200;
   3: GPS1.PortBaud := 2400;
   4: GPS1.PortBaud := 4800;
   5: GPS1.PortBaud := 9600;
   6: GPS1.PortBaud := 19200;
 end;
end;
procedure TForm1.rgDatabitsClick(Sender: TObject);
begin
 case rgDatabits.ItemIndex of
   0: GPS1.PortDatabits := 7;
   1: GPS1.PortDatabits := 8;
```

```
 end;
end;
procedure TForm1.rgStopbitsClick(Sender: TObject);
begin
 case rgDatabits.ItemIndex of
  0: GPS1.PortStopBits := 1;
  1: GPS1.PortStopBits := 2;
 end;
end;
procedure TForm1.rgParityClick(Sender: TObject);
begin
 case rgParity.ItemIndex of
  0: GPS1.PortParity := cpNone;
  1: GPS1.PortParity := cpOdd;
  2: GPS1.PortParity := cpEven;
  3: GPS1.PortParity := cpMark;
  4: GPS1.PortParity := cpSpace;
 end;
end;
procedure TForm1.GPS1NavigationInfo(Sender: TObject; Warning,
 Arrived: Boolean; CTE: Double; OriginWaypoint, DestWaypoint: String;
 DestLat, DestLon, RangeDest, BearingDest, Velocity: Double);
begin
 if Warning then
  edRMB1.Text := 'Yes'
 else
  edRMB1.Text := 'No';
 if Arrived then
  edRMB6.Text := 'Yes'
 else
```

```
  edRMB6.Text := 'No';
 edRMB9.Text := Format('%.2f',[CTE]);
 edRMB2.Text := Format('%.4f',[DestLat]);
 edRMB3.Text := Format('%.4f',[DestLon]);
 edRMB4.Text := Format('%.2f',[BearingDest]);
 edRMB5.Text := Format('%.2f',[RangeDest]);
 edRMB7.Text := OriginWaypoint;
 edRMB8.Text := DestWaypoint;
 edRMB10.Text := Format('%.2f',[Velocity]);
end;
procedure TForm1.cbDTRClick(Sender: TObject);
begin
 GPS1.SetDTR(cbDTR.Checked);
 GPS1.SetRTS(cbRTS.Checked);
end;
procedure TForm1.btnStopAutodetectClick(Sender: TObject);
begin
 btnStopAutodetect.Enabled := False;
 lblAutoDetect.Caption := 'Stopping...';
 GPS1.StopAutodetect;
 lblStatus.Caption := '';
 lblAutoDetect.Caption := 'Result';
 btnAutodetect.Enabled := True;
end;
procedure TForm1.GPS1Status(Sender: TObject; Status: TNMEAStatus);
begin
 case Status of
  nsNoPort:
   lblStatus.Caption := 'No comm port';
  nsNoComm:
```

```
    lblStatus.Caption := 'No communication';
  nsComm:
    lblStatus.Caption := 'Communication';
  nsSentence:
    lblStatus.Caption := 'NMEA';
  nsSentenceGPS:
    lblStatus.Caption := 'NMEA GPS';
 end;
 if GPS1.AutodetectOn then
  lblAutoDetect.Caption := 'Checking COM' + IntToStr(Ord(GPS1.Port) + 1) + ', ' +
IntToStr(GPS1.PortBaud);
end;
procedure TForm1.Timer1Timer(Sender: TObject);
begin
    ZQuery1.Close;
    ZQuery1.SQL.Clear;
    ZQuery1.SQL.add('INSERT INTO markers
(suhu,Kelembaban,latitude,longitude,lingkungan,dewpoint,waktu,id) values
('''+txtsuhu.text+''','''+txtkelembaban.Text+''','''+edGGA2.Text+''','''+edGGA3.text+''','''+cuaca.t
ext+''','''+dewpoint.text+''','''+label61.Caption+''','''+edit1.text+''')');
    ZQuery1.ExecSQL;
end;
procedure TForm1.Button6Click(Sender: TObject);
begin
    Timer1.Enabled := True;
end;
procedure TForm1.btnsettingClick(Sender: TObject);
begin
CP1.ShowSetupDialog;
end;
procedure TForm1.btnconnectClick(Sender: TObject);
begin
```

```
if  CP1.Connected = true then
        begin
        CP1.Close;
    //     LblStatusCom.Caption := CP1.Port + ' Disconnected';
        BtnConnect.Caption := 'Connect';
        BtnSetting.Enabled := true;

        end
      else
        begin
        CP1.Open;
       // LblStatusCom.Caption := CP1.Port + '   Connected';
        BtnConnect.Caption := 'Disconnect';
        BtnSetting.Enabled := False;
//         timer1.Enabled:=true;
        Timer2.Enabled := True;
        DataTrans := $01;
        end;
end;
procedure TForm1.IntSerialRX(Sender: TObject; Count: Integer);
begin
      CP1.Read(DataSerial,1);
      FlagRX := FlagRX + 1;
      if (FlagRX = 1) then
      begin
      DataSerialTemp := DataSerial;
      end;
      if (FlagRX = 2) then
      begin
      DataSerialTemp := DataSerialTemp SHL 8;
```

```
      DataSerialTemp := DataSerialTemp OR DataSerial;
      FlagRX := 0;
      DataSHT := DataSerialTemp;
//Konversi Temperature dari SHT11
//  Celcius = (SOt * d2) + d1
//    dengan d2 = 0.01 (14-bit) dan d1 = -40 (VDD = 5Volt)
//  Celcius = (ADC_Temp * 0.01) - 40   -- dikalikan dengan 100
//  Celcius = (ADC_Temp - 4000)
//Catatan: Faktor konversi pada rumus dikalikan dengan 100 untuk
//       memperoleh hasil pengukuran temperature per-100 derajat (0,01)
//       dengan adanya pembulatan
        if (DataTrans = 1) then
        begin
        DataSHT := (DataSHT-4000) / 100;
        txtsuhu.Text := CurrToStr(DataSHT);
        //LblTemp.Caption := CurrToStr(DataSHT);
        DataTrans := 2;
        end
//Konversi Relative Humidity dari SHT11
//  Relatitive Humidity (linear)
//    RHLin = c1 + (c2 * SORH) + (c3 * SORH^2)
//  Untuk pengukuran 12-bit: c1 = -4; c2 = 0,0405 dan c3 = -2,8*10^-6
//    RHLin = (ADC_RH * 0,0405) - (ADC_RH^2 * 0,0000028) - 4
        else if (DataTrans = 2) then
        begin
        Temp := DataSHT;
        DataSHT := (Temp * 0.0405) - (Temp * Temp * 0.0000028) - 4;
        txtkelembaban.Text  := CurrToStr(DataSHT);
        //LblHumi.Caption := CurrToStr(DataSHT);
        DataTrans := 1;
```

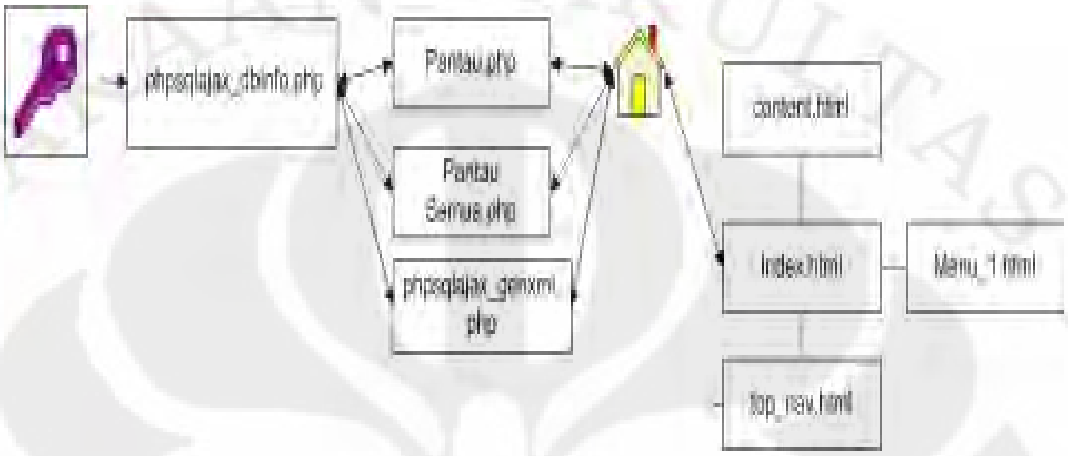
```
        end
        else DataTrans := 1;
        Timer2.Enabled := True;
        end;
end;
procedure TForm1.IntTim2(Sender: TObject);
var a,b,c,x,y : real;
begin
      if CP1.Connected = True then
      begin
        CP1.Write(DataTrans,1);
        FlagRX := 0;
        x := strtofloat(txtsuhu.Text);//:=strtofloat(x);
        y := strtofloat(txtkelembaban.Text);// := strtofloat(y);
        a := 100-y;
        b := a/5;
        c := x-b;
        dewpoint.Text := floattostr(c);
        if x-c < 2.5 then
        cuaca.Text := 'Berkabut';
        if x-c > 2.5 then
        cuaca.Text := 'Tidak Berkabut'
        else
        cuaca.Text := 'Tidak didefiniskan';
        end;
      Timer2.Enabled := False;
end;
procedure TForm1.waktusekarangTimer(Sender: TObject);
begin
label61.Caption := timetostr(time);
```

```
end;

procedure TForm1.Timer3Timer(Sender: TObject);

begin

    ZQuery1.Close;

    ZQuery1.SQL.Clear;

    ZQuery1.SQL.add('UPDATE markers set suhu ='''+txtsuhu.text+''',Kelembaban =
'''+txtkelembaban.Text+''',latitude = '''+edGGA2.Text+''',longitude
='''+edGGA3.text+''',lingkungan = '''+edGGA3.text+''',dewpoint ='''+dewpoint.text+'''where id =
'''+id.Text+''');

    ZQuery1.ExecSQL;

end;

procedure TForm1.Button4Click(Sender: TObject);

begin

Timer3.Enabled := True;

end;

procedure TForm1.Button5Click(Sender: TObject);

begin

Application.Minimize;

end;

end.
```

LAMPIRAN 9

SOURCE CODE APLIKASI WEBSITE

Komponen Aplikasi digambarkan pada gambar berikut ini :



1. phpsqlajax_dbinfo.php

```
<?
 $username = 'root';
 $password = '';
 $database = 'kml';
 $server = 'localhost'
?>
```

2. Pantau.php

```
<?php
require('phpsqlajax_dbinfo.php');


 // Opens a connection to a MySQL server.
$connection = mysql_connect ($server, $username, $password);
if (!$connection)
{
  die('Not connected : ' . mysql_error());
}
```

```
// Sets the active MySQL database.
$db_selected = mysql_select_db($database, $connection);
if (!$db_selected)
{
  die ('Can\'t use db : ' . mysql_error());
}

 // Pilih Akhir Tabel
 $query = 'SELECT * FROM markers ORDER BY id DESC LIMIT 0,1';
 $result = mysql_query($query);
 if (!$result)
 {
  die('Invalid query: ' . mysql_error());
 }

// Creates an array of strings to hold the lines of the KML file.
$kml = array('<?xml version="1.0" encoding="UTF-8"?>');
$kml[] = '<kml xmlns="http://earth.Google.com/kml/2.1">';
$kml[] = ' <Document>';
$kml[] = ' <Style id="pantau">';
$kml[] = ' <IconStyle id="pantau">';
$kml[] = ' <Icon>';
$kml[] = ' <href>http://maps.Google.com/mapfiles/kml/paddle/ylw-stars.png</href>';
$kml[] = ' </Icon>';
$kml[] = ' </IconStyle>';
$kml[] = ' </Style>';

// Iterates through the rows, printing a node for each row.
while ($row = @mysql_fetch_assoc($result))
{
```

```
// data suhu dan kelembaban


 $kml[] = ' <Placemark id="placemark' . $row['id'] . '">';

 $kml[] = ' <Style>';

 $kml[] = ' <IconStyle>';

 $kml[] = ' <Icon>';

 $kml[] = ' <href>http://maps.Google.com/mapfiles/kml/paddle/ylw-stars.png</href>';

 $kml[] = ' </Icon>';

 $kml[] = ' </IconStyle>';

 $kml[] = ' </Style>';

 $kml[] = ' <name>Pantau</name>';

 $kml[] = ' <description>  <![CDATA[<TABLE border="1"><TR><TD>Suhu Udara</TD><TD>' .
htmlentities($row['suhu']) . ' derajat celcius</TD></TR><TR><TD>Kelembaban</TD><TD>' .
htmlentities($row['Kelembaban']) . ' persen</TD></TR><TR><TD>Garis Lintang</TD><TD>' .
htmlentities($row['longitude']) . '</TD></TR><TR><TD>Garis Bujur</TD><TD>' .
htmlentities($row['latitude']) . '</TD></TR><TR><TD>DewPoint</TD><TD>' .
htmlentities($row['dewpoint']) . ' derajat celcius</TD></TR><TR><TD>Parameter
Lingkungan</B></TD><TD>' . htmlentities($row['lingkungan']) . '
</TD></TR></TD></TR></TABLE>]]></description>';

 $kml[] = ' <Point>';

 $kml[] = ' <coordinates>' . $row['longitude'] . ',' . $row['latitude'] . '</coordinates>';

 $kml[] = ' </Point>';

 $kml[] = ' </Placemark>';

    }

// End XML file

 $kml[] = ' </Document>';


$kml[] = '</kml>';

$kmlOutput = join("\n", $kml);

header('Content-type: application/vnd.Google-earth.kml+xml');

echo $kmlOutput;

?>
```

3. Pantau Semua.php

```
<?php
require('phpsqlajax_dbinfo.php');


 // Opens a connection to a MySQL server.
$connection = mysql_connect ($server, $username, $password);
if (!$connection)
{
  die('Not connected : ' . mysql_error());
}


// Sets the active MySQL database.
$db_selected = mysql_select_db($database, $connection);
if (!$db_selected)
{
  die ('Can\'t use db : ' . mysql_error());
}


 // Pilih Akhir Tabel
 $query = 'SELECT * FROM markers ORDER BY id DESC LIMIT 0,1';
 $result = mysql_query($query);
 if (!$result)
 {
  die('Invalid query: ' . mysql_error());
 }


// Creates an array of strings to hold the lines of the KML file.
$kml = array('<?xml version="1.0" encoding="UTF-8"?>');
$kml[] = '<kml xmlns="http://earth.Google.com/kml/2.1">';
$kml[] = ' <Document>';
```

```
$kml[] = ' <Style id="pantau">';

$kml[] = ' <IconStyle id="pantau">';

$kml[] = ' <Icon>';

$kml[] = ' <href>http://maps.Google.com/mapfiles/kml/paddle/ylw-stars.png</href>';

$kml[] = ' </Icon>';

$kml[] = ' </IconStyle>';

$kml[] = ' </Style>';


// Iterates through the rows, printing a node for each row.

while ($row = @mysql_fetch_assoc($result))

{

// data suhu dan kelembaban


 $kml[] = ' <Placemark id="placemark' . $row['id'] . '">';

 $kml[] = ' <Style>';

 $kml[] = ' <IconStyle>';

 $kml[] = ' <Icon>';

 $kml[] = ' <href>http://maps.Google.com/mapfiles/kml/paddle/ylw-stars.png</href>';

 $kml[] = ' </Icon>';

 $kml[] = ' </IconStyle>';

 $kml[] = ' </Style>';

 $kml[] = ' <name>Pantau</name>';

 $kml[] = ' <description>  <![CDATA[<TABLE border="1"><TR><TD>Suhu Udara</TD><TD>' .
htmlentities($row['suhu']) . ' derajat celcius</TD></TR><TR><TD>Kelembaban</TD><TD>' .
htmlentities($row['Kelembaban']) . ' persen</TD></TR><TR><TD>Garis Lintang</TD><TD>' .
htmlentities($row['longitude']) . '</TD></TR><TR><TD>Garis Bujur</TD><TD>' .
htmlentities($row['latitude']) . '</TD></TR><TR><TD>DewPoint</TD><TD>' .
htmlentities($row['dewpoint']) . ' derajat celcius</TD></TR><TR><TD>Parameter
Lingkungan</B></TD><TD>' . htmlentities($row['lingkungan']) . '
</TD></TR></TD></TR></TABLE>]]></description>';

 $kml[] = ' <Point>';

 $kml[] = ' <coordinates>' . $row['longitude'] . ',' . $row['latitude'] . '</coordinates>';

 $kml[] = ' </Point>';
```

```
 $kml[] = ' </Placemark>';
    }
// End XML file
 $kml[] = ' </Document>';

$kml[] = '</kml>';
$kmlOutput = join("\n", $kml);
header('Content-type: application/vnd.Google-earth.kml+xml');
echo $kmlOutput;
?>
```

4. Pantau Semua.php

```
<?php
require("phpsqlajax_dbinfo.php");

function parseToXML($htmlStr)
{
$xmlStr=str_replace('<','&lt;',$htmlStr);
$xmlStr=str_replace('>','&gt;',$xmlStr);
$xmlStr=str_replace('"','&quot;',$xmlStr);
$xmlStr=str_replace("'",'&#39;',$xmlStr);
$xmlStr=str_replace("&",'&amp;',$xmlStr);
return $xmlStr;
}

// Opens a connection to a MySQL server
$connection=mysql_connect (localhost, $username, $password);
if (!$connection) {
  die('Not connected : ' . mysql_error());
```

```
}

// Set the active MySQL database
$db_selected = mysql_select_db($database, $connection);
if (!$db_selected) {
  die ('Can\'t use db : ' . mysql_error());
}

// Select all the rows in the markers table
 $query = 'SELECT * FROM markers ORDER BY id DESC LIMIT 0,1';
$result = mysql_query($query);
if (!$result) {
  die('Invalid query: ' . mysql_error());
}

header("Content-type: text/xml");

// Start XML file, echo parent node
echo '<markers>';

// Iterate through the rows, printing XML nodes for each
while ($row = @mysql_fetch_assoc($result)){
  // ADD TO XML DOCUMENT NODE
  echo '<marker ';
  //echo '<![CDATA[<TABLE border="1"><TR><TD><B>Suhu Udara</B></TD><TD><B>' .
htmlentities($row['suhu']) . ' derajat celcius</B></TD></TR><TR><TD>Kelembaban</TD><TD>'
. htmlentities($row['Kelembaban']) . ' persen</TD></TR><TR><TD>Garis Lintang</TD><TD>' .
htmlentities($row['longitude']) . '</TD></TR><TR><TD>Garis Bujur</TD><TD>' .
htmlentities($row['latitude']) . '</TD></TR><TR><TD><B>Kecepatan</B></TD><TD><B>' .
htmlentities($row['kecepatan']) . ' kilo meter per jam</B></TD></TR></TABLE>]]>';
  echo 'suhu="' . parseToXML($row['suhu']) . '" ';
  echo 'Kelembaban="' . parseToXML($row['Kelembaban']) . '" ';
```

```
 echo 'latitude="' . $row['latitude'] . '" ';

 echo 'longitude="' . $row['longitude'] . '" ';

 echo 'type="' . $row['type'] . '" ';

 echo '/>';

}


// End XML file

echo '</markers>';


?>
```

4. Index.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

<html>

<head>

<title>Sistem Pemantauan Kapal dan Parameter Cuaca Laut BERBASIS Google Earth dan Google Maps</title>

</head>

<!--


Note the following:

1. Each frame has it's own 'frame' tag.

2. Each frame has a name (eg, name="menu"). This is used for when you need to load one frame from another. For example, your left frame might have links that, when clicked on, loads a new page in the right frame. This is acheived by using 'target="content"' within your links/anchor tags.

3. Each 'frame' tag has a 'src' attribute. This is where you specify the name of the file to be loaded into that frame when the page first loads.

4. You can change the size of the frames by changing the value of the 'cols' and/or 'rows' attribute. A value of "200" sets the frame at 200 pixels. An asterisk (*) specifies that the frame should use up whatever space is left over from the other frames. You can also use percentage values if desired (i.e. 20%,80%)

5. To specify a border, set 'frameborder' and 'border' to "1". I included both attributes for maximum browser compatibility.
```

6. The 'framespacing' attribute doesn't work in all browsers, but you can set any numeric value you like here.

7. To learn more about HTML frames, check out: http://www.quackit.com/html/tutorial/html_frames.cfm

-->

<frameset rows="100,*" frameborder="0" border="0" framespacing="0">

<frame name="topNav" src="top_nav.html">

<frameset cols="200,*" frameborder="0" border="0" framespacing="0">

<frame name="menu" src="menu_1.html" marginheight="0" marginwidth="0" scrolling="auto" noresize>

<frame name="content" src="phpsqlajax_map.htm" marginheight="0" marginwidth="0" scrolling="auto" noresize>

<noframes>

** This section (everything between the 'noframes' tags) will only be displayed if the users' browser doesn't support frames. You can provide a link to a non-frames version of the website here. Feel free to use HTML tags within this section. **

</noframes>

</frameset>

</frameset>

</html>

5. Top_Nav.html

```
<html>
<head>
<title>HTML Frames Example - Top Nav</title>
<style type="text/css">
body {
        font-family:verdana,arial,sans-serif;
        font-size:14pt;
        margin:10px;
        background-color:#a08029;
```

```
        }
</style>
</head>
<body>
<h3>Sistem Pemantauan Cuaca BERBASIS Google Earth dan Google Maps</h3>
</body>
</html>
```

6. Menu-1.html

```
<html>
<head>
<title>HTML Frames Example - Menu 2</title>
<style type="text/css">
body {
        font-family:verdana,arial,sans-serif;
        font-size:9pt;
        margin:10px;
        background-color:#ff6600;
        }
</style>
</head>
<body>
<h3>Menu Utama</h3>
<p><a href="pantau.php" target="content">Unduh File KML Pemantauan Tunggal</a></p>
<p><a href="pantau semua.php" target="content">Unduh File KML Pemantauan</a></p>
<p>Arahkan Network Link Google Earth Pada Alamat : </p>
<p>http://152.118.24.12/skripsi/phpsql_genkml.php</p>
<p> </p>
<p> </p>
</body>
```

```
</html>
```

6. Content.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">


<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8"/>
        <meta http-equiv="refresh" content="8">
  <title>Peta Informasi Sensor</title>
        <style type="text/css">
   v\:* {
    behavior:url(#default#VML);
   }
   #MapBuilder {font: normal small verdana, arial, helvetica, sans-serif; font-size: 10pt; margin: 0px;}
   #MapBuilder a {text-decoration: none; color: #0066CC; background-color: transparent;}
   #MapBuilder a:hover {color: #F60; background-color: transparent;}
   #MapBuilder h1 {font-weight: bold; font-size: 16pt; color: #369; background-color: transparent; border-bottom: 2px solid #369;}
   #MapBuilderIW { width: 350px; height: 200px;}
   #MapBuilderIWContent {height: 120px; overflow:auto;}
   #MapBuilderIW h1 {font-weight: bold; font-size: 12pt; color: #369; background-color: transparent; border-bottom: 2px solid #369;}
   #MapBuilderIWFooter {margin-top: 5px; font-family: Arial, Helvetica, sans-serif; font-weight: normal; font-size: 8pt; }
   #MapBuilderIWFooter h1 { margin: 2px 0px 4px font-weight: normal; font-size: 8pt; color: #369; background-color: transparent; border-top: 2px solid #369; border-bottom: 0px;}

   #MapBuilderSideBar {float:left; margin-left: 10px; }

  </style>
```

```
        <center>

  <script
src="http://maps.Google.com/maps?file=api&v=2&key=ABQIAAAAq3pCxptG2ixSNRODS1WachT2yXp_ZAY8_ufC3CFXhHIE1NvwkxQjKEoJgDqLDGcGdXceQNQhCduFKA"

      type="text/javascript"></script>


  <script type="text/javascript">

  //<![CDATA[


  var iconBlue = new GIcon();

  iconBlue.image = 'http://labs.Google.com/ridefinder/images/mm_20_blue.png';

  iconBlue.shadow = 'http://labs.Google.com/ridefinder/images/mm_20_shadow.png';

  iconBlue.iconSize = new GSize(12, 20);

  iconBlue.shadowSize = new GSize(22, 20);

  iconBlue.iconAnchor = new GPoint(6, 20);

  iconBlue.infoWindowAnchor = new GPoint(5, 1);


  var iconRed = new GIcon();

  iconRed.image = 'http://labs.Google.com/ridefinder/images/mm_20_red.png';

  iconRed.shadow = 'http://labs.Google.com/ridefinder/images/mm_20_shadow.png';

  iconRed.iconSize = new GSize(12, 20);

  iconRed.shadowSize = new GSize(22, 20);

  iconRed.iconAnchor = new GPoint(6, 20);

  iconRed.infoWindowAnchor = new GPoint(5, 1);


  var customIcons = [];

  customIcons["restaurant"] = iconBlue;

  customIcons["bar"] = iconRed;


  function load() {

   if (GBrowserIsCompatible()) {
```

```
    var map = new GMap2(document.getElementById("map"));
    map.addControl(new GSmallMapControl());
    map.addControl(new GMapTypeControl());
    map.setCenter(new GLatLng(-6.369700, 106.825600), 14);

    GDownloadUrl("phpsqlajax_genxml.php", function(data) {
     var xml = GXml.parse(data);
     var markers = xml.documentElement.getElementsByTagName("marker");
     for (var i = 0; i < markers.length; i++) {
      var suhu = markers[i].getAttribute("suhu");
      var Kelembaban = markers[i].getAttribute("Kelembaban");
      var type = markers[i].getAttribute("type");
      var point = new GLatLng(parseFloat(markers[i].getAttribute("latitude")),
                              parseFloat(markers[i].getAttribute("longitude")));
      var marker = createMarker(point, suhu, Kelembaban, type);
      map.addOverlay(marker);
     }
    });
   }
  }

  function createMarker(point, suhu, Kelembaban, type) {
   var marker = new GMarker(point, customIcons[type]);
   var html = "<b>" + suhu + "</b> <br/>" + Kelembaban;
   GEvent.addListener(marker, 'click', function() {
    marker.openInfoWindowHtml(html);
   });
   return marker;
  }
  //]]>
```

```
 </script>
</center>
 </head>

 <body onload="load()" onunload="GUnload()">
  <div id="map" style="width: 1100px; height: 550px"></div>
 </body>
</html>
```