PERANCANGAN PERANGKAT LUNAK *DRIVE TEST* BERBASIS ANDROID UNTUK ANALISIS KUALITAS *VOICE CALL*

SKRIPSI

Agung Budiargo
NIM 091910201095

PROGRAM STUDI STRATA 1 TEKNIK ELEKTRO
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS JEMBER
2013

PENGESAHAN

Skripsi dengan judul : **"*Perancangan Perangkat Lunak Drive Test Berbasis Android Untuk Analisis Kualitas Voice Call*"** telah diuji dan disahkan oleh Jurusan Teknik Elektro, Fakultas Teknik, Universitas Jember pada :

Hari : Selasa

Tanggal : 24 September 2013

Tempat : Fakultas Teknik Universitas Jember

Tim Penguji

Pembimbing Utama,	Pembimbing Anggota,
Ike Fibriani, S.T.,M.T.	**M. Agung Prawira N, S.T.,M.T.**
NRP : 760011391	**NIP : 19871217 201212 1 003**

Mengetahui

Penguji I,	Penguji II,
Catur Suko Suwono, S.T	**Bambang Supeno, S.T.,M.T.**
NIP : 19680119 199702 1 001	**NIP : 19690630 199512 1 001**

Mengesahkan,

Dekan Fakultas Teknik,

Universitas Jember.

Ir. Widyono Hadi, M.T.

NIP : 19610414 198902 1 001

PERANCANGAN PERANGKAT LUNAK *DRIVE TEST* BERBASIS ANDROID UNTUK ANALISIS KUALITAS *VOICE CALL*

Agung Budiargo
Mahasiswa Jurusan Teknik Elektro.
Fakultas Teknik, Universitas Jember.

ABSTRAK

Drive test adalah salah satu langkah awal dalam proses optimasi jaringan yang bertujuan untuk mengumpulkan data-data pengukuran pada area yang kurang optimal. *Drive test* membutuhkan banyak alat seperti laptop terinstal *software* TEMS, *handphone*, kabel data, GPS dan biasanya menggunakan mobil. Penelitian ini bertujuan untuk merancang aplikasi android yang bisa digunakan untuk *drive test* kualitas *voice call* tanpa memerlukan banyak alat dan data hasil *drive test* bisa digunakan untuk optimasi jaringan pada area yang kurang optimal. Hasil dari analisis data aplikasi *drive test* yang diberi nama Net Info menunjukkan bahwa nilai RSCP pada *cluster* Uluwatu masih buruk dengan prosentase 20,31%, untuk Ec/No sudah cukup baik dengan prosentase Ec/No buruk hanya 1,58% dan hanya terjadi 1 kali *block call*. Perbandingan *drive test* menggunakan aplikasi Net Info dengan menggunakan *software* TEMS Investigations jika dilihat dari hasil grafik maupun *plotting* data hasilnya mirip, namun untuk tingkat ketelitian dari Net Info masih sangat kecil jika dibandingkan dengan *software* TEMS.

Kata kunci : *Drive test*, Ec/No, Net Info, RSCP.

DESIGN OF DRIVE TEST SOFTWARE BASED ON ANDROID FOR VOICE CALL QUALITY ANALYSIS

Agung Budiargo

College Student of Department of Electrical Engineering.

Engineering Faculty, Jember University.

ABSTRACT

Drive test is one of the first steps in the process of network optimization to collect measurement data in areas that are less optimal. Drive test requires a lot of tools like TEMS software installed laptops, mobile phones, data cable, GPS and usually use the car. This research is to design a android application that can be used to drive test the quality of voice calls without needing a lot of tools and drive test result data can be used for network optimization in areas that are less optimal. Results of the data analysis application called test drives showed that the value of the Net info RSCP at Uluwatu cluster is still bad with the percentage of 20.31%, for Ec / No percentage is pretty good with bad Ecno only 1.58% and occurs only 1 time block call. Comparison test drive using Info Net application using TEMS Investigations software if seen from the graph plotting the data and the result is similar, but for the level of accuracy of the Net info is still very small when compared to the TEMS software.

Keyword : *Drive test, Ec/No, Net Info, RSCP.*

RINGKASAN

Perancangan Perangkat Lunak *Drive Test* Berbasis Android Untuk Analisis Kualitas *Voice Call*; Agung Budiargo, 091910201095; 2013; 61 halaman; Jurusan Teknik Elektro Fakultas Teknik Universitas Jember.

Perkembangan teknologi komunikasi seluler semakin meningkat. Akan tetapi, masih ditemukan berbagai permasalahan seperti kualitas panggilan yang tidak bagus. Indikator yang menunjukkan terjadinya permasalahan yang berkaitan dengan kualitas panggilan antara lain terjadinya *dropped call* dan *blocked call.* Untuk mengetahui permasalahan pada jaringan, maka dilakukan *drive test. Drive test* adalah salah satu langkah awal dalam proses optimasi jaringan yang bertujuan untuk mengumpulkan data-data pengukuran pada suatu area yang kurang optimal. Namun masih terdapat banyak kendala dalam melakukan *drive test,* antara lain membutuhkan banyak alat seperti laptop yang terinstal *software* TEMS, *handphone,* kabel data, GPS, dan biasanya dilakukan dengan menggunakan mobil, sehingga kesulitan ketika dilakukan di jalan-jalan sempit maupun jalan yang padat lalu lintas. Berdasarkan uraian tersebut, dibutuhkan perangkat *drive test* yang tidak membutuhkan banyak alat sehingga tidak harus menggunakan mobil tetapi bisa menggunakan sepeda motor ataupun jalan kaki, dan perangkat yang bisa digunakan dalam waktu yang lama. Pada penelitian ini akan dilakukan perancangan perangkat lunak *drive test* berbasis android yang bisa digunakan untuk menganalisa kualitas *voice call*. Sehingga proses *drive test* tidak lagi membutuhkan banyak alat, hanya membutuhkan *smartphone* android dan tidak membutuhkan GPS tambahan karena *smartphone* android sudah ada GPS. Dengan demikian *drive test* bisa dilakukan dengan waktu yang jauh lebih lama serta tidak harus menggunakan alat transportasi mobil.

Proses perancangan aplikasi android dengan menggunakan Android Studio sebagai *software editor* dan MapInfo Professional 11 yang digunakan untuk analisis datanya. Setelah proses perancangan perangkat lunak *drive test* yang diberi nama Net

Info ini selesai, maka dilakukan proses pengambilan data pada area *cluster* Uluwatu. Data yang didapatkan hasil dari *drive test* Net Info selanjutnya dianalisis berdasarkan nilai RSCP, Ec/No, *scrambling code, block call* dan *drop call*. Pada tahap akhirnya data akan dibandingkan dengan hasil *drive test* menggunakan TEMS Investigations. Dari hasil analisis data Net Info pada *cluster* Uluwatu, nilai RSCP tergolong buruk dimana prosentase RSCP *low coverage* mencapai 20,31%. Dalam hal ini perlu dilakukan optimasi lagi pada area tersebut untuk meningkatkan *coverage* dari jaringan. Untuk nilai Ec/No sudah tergolong cukup baik dengan prosentase nilai buruk hanya 1,58% dan hanya terjadi 1 kali *block call*. Perbandingan *drive test* menggunakan aplikasi Net Info dengan menggunakan *software* TEMS Investigations jika dilihat dari hasil grafik maupun *plotting* data hasilnya mirip, namun untuk tingkat ketelitian dari Net Info masih sangat kecil jika dibandingkan dengan *software* TEMS yaitu hanya 1 : 32.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB 1. PENDAHULUAN

Android merupakan perangkat cerdas bersifat *opensource* yang berdampak pada meningkatnya jumlah pengguna maupun pengembang aplikasi secara *continue* dan signifikan. Alasan ini menjadi peluang yang sangat bagus bagi praktisi IT, mahasiswa, bahkan seorang penghobi untuk berpartisipasi mengembangkan aplikasi android (Huda, 2012).

1.1. Latar Belakang

Perkembangan teknologi komunikasi seluler terus menunjukkan peningkatan baik dari segi produk maupun layanan. Setiap penyedia (*provider*) jaringan komunikasi seluler, berusaha memberikan pelayanan yang terbaik untuk pelanggannya. Akan tetapi, masih ditemukan berbagai permasalahan pada jaringan tersebut. Salah satu permasalahannya ialah kualitas panggilan yang tidak bagus. Indikator yang menunjukkan terjadinya permasalahan yang berkaitan dengan kualitas panggilan antara lain terjadinya *dropped call* dan *blocked call* (Novrizal, 2011).

Untuk mengetahui permasalahan pada jaringan, maka dilakukan survei lapangan dan menerima keluhan dari pelanggan. Dari keluhan pelanggan tersebut kemudian dilakukan *drive test. Drive test* adalah salah satu langkah awal dalam proses optimasi jaringan yang bertujuan untuk mengumpulkan data-data pengukuran pada suatu area yang kurang optimal. Data tersebut dapat dipergunakan untuk mengidentifikasi masalah-masalah jaringan seperti level sinyal yang lemah, kualitas sinya yang buruk, dan sebagainya (Siboro, 2011). Namun masih terdapat banyak kendala dalam melakukan *drive test,* antara lain membutuhkan banyak alat seperti laptop yang terinstal *software* TEMS, *handphone,* kabel data, GPS, dan luarannya diolah dengan *software* MapInfo. Karena proses *drive test* dilakukan dengan menggunakan laptop, maka hanya mampu dilakukan selama 3 sampai 4 jam saja. Selain itu *drive test* biasanya dilakukan dengan menggunakan mobil, sehingga kesulitan ketika dilakukan di jalan-jalan sempit maupun jalan yang padat lalu lintas.

Berdasarkan uraian tersebut, dibutuhkan perangkat *drive test* yang tidak membutuhkan banyak alat sehingga tidak harus menggunakan mobil tetapi bisa menggunakan sepeda motor ataupun jalan kaki, dan perangkat yang bisa digunakan dalam waktu yang lama.

Pemilihan telepon seluler *platform* berbasis android untuk salah satu pengembangan aplikasi selain bersifat *opensource* dan lebih mudah dalam pengoperasiannya, sifat dari telepon seluler yang fleksibel menjadi salah satu alasannya. Sekarang ini kebutuhan masyarakat akan *smartphone* sudah semakin meningkat. Hingga saat ini tercatat perkembangan *smartphone* berbasis Android meningkat drastis dengan melihat data dari Telkomsel pada desember 2011 yang sudah mencapai 900 ribu *user* (Arifin, 2012). Selain itu banyak vendor-vendor yang menggunakan sistem operasi android untuk *smartphone* mereka, seperti HTC, Motorola, Samsung, LG, HKC, Huawei, Archos, Webstation Camangi, Dell, Nexus, SciPhone, WayteQ, Sony Ericsson, Acer, Philips, T-Mobile, Nexian, IMO, Asus, dan masih banyak lagi vendor-vendor lain (Supardi, 2011). Penggunanya juga bervariasi mulai dari anak-anak sampai orang dewasa. Penggunaan telepon seluler selain sebagai media komunikasi, juga bisa dijadikan sebagai sarana informasi. Beragam aplikasi telah banyak berjalan dalam telepon seluler khusunya *platform* android.

Untuk itulah pada Tugas Akhir ini akan dilakukan perancangan perangkat lunak *drive test* berbasis android yang bisa digunakan untuk menganalisa kualitas *voice call*. Sehingga proses *drive test* tidak lagi membutuhkan banyak alat, hanya membutuhkan *smartphone* dengan *platform* android dan aplikasi *drive test* serta tidak membutuhkan GPS tambahan karena *smartphone* android sudah ada GPS. Dengan demikian *drive test* bisa dilakukan dengan waktu yang jauh lebih lama serta tidak kesulitan ketika dilakukan di jalan-jalan sempit maupun jalan yang padat lalu lintas karena tidak harus menggunakan alat transportasi mobil.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, permasalahan yang muncul dalam perancangan alat ini antara lain:

1. Bagaimana hasil analisis performansi jaringan 3G WCDMA menggunakan perangkat lunak *drive test.*
2. Bagaimana keefektifan penggunaan GPS *smartphone* untuk *drive test*.
3. Bagaimana perbandingan data *drive test* menggunakan perangkat lunak dengan *drive test* menggunakan TEMS.

1.3. Batasan Masalah

Batasan permasahan untuk menghindari pembahasan yang terlalu luas, menyederhanakan dan memperjelas, maka dalam skripsi ini hanya akan memberikan pembahasan sebagai berikut:

1. Perancangan perangkat lunak *drive test* berbasis android untuk minimum android versi 2.2 (Froyo: *Frozen Yoghurt*).
2. Perancangan perangkat lunak hanya difokuskan untuk analisis kualitas *voice call* jaringan WCDMA.
3. *Software editor* untuk membuat aplikasi android adalah Android Studio.

1.4. Tujuan

Perancangan perangkat lunak ini memiliki tujuan sebagai berikut:

1. Untuk menganalisis performansi jaringan WCDMA dengan menggunakan data hasil dari perangkat lunak *drive test.*
2. Untuk memfungsikan GPS *smartphone* untuk *drive test.*
3. Untuk menghasilkan perangkat lunak atau aplikasi android yang bisa digunakan untuk *drive test* analisis kualitas panggilan suara jaringan WCDMA.

1.5. Manfaat

Hasil dari perancangan perangkat lunak ini diharapkan memiliki manfaat sebagai berikut :

1. Dari hasil *drive test* maka dapat dilakukan proses optimasi untuk meningkatkan performansi jaringan WCDMA khususnya kualitas *voice call*
2. Tidak membutuhkan GPS tambahan dalam melakukan *drive test* karena sudah memfungsikan GPS yang ada pada *smartphone* android untuk *drive test.*
3. Perangkat lunak ini dapat menggantikan perangkat *drive test* yang digunakan saat ini, sehingga tidak membutuhkan banyak alat lagi hanya membutuhkan *smartphone* yang terinstal perangkat lunak *drive test.*

1.6. Sistematika Penulisan

Secara garis besar penyususnan proposal penelitian tugas akhir ini adalah sebagai berikut:

BAB 1 PENDAHULUAN

Berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat penulisan juga sistematika penulisan yang digunakan. Bab ini diharapkan dapat memberi gambaran awal tentang studi analisis yang akan dilakukan ini.

BAB 2 TINJAUAN PUSTAKA

Berisi penjelasan tentang materi yang berisi teori untuk mendukung perancangan perangkat lunak pada penelitian tugas akhir ini. Materi yang akan dibahas di bab ini diantaranya dasar pemrograman, sistem komunikasi bergerak, metode *drive test* dan parameter analisis kualitas *voice call*.

BAB 3 METODOLOGI PENELITIAN

Menjelaskan tentang waktu dan tempat penelitian, metode kajian yang digunakan untuk menyelesaikan skripsi, sistematika analisis, dan prosesnya dalam bentuk diagram alur.

BAB 4 PEMBAHASAN

Menjelaskan tentang perancangan perangkat lunak, analisis data hasil *drive test* dan perbandingan perngakat lunak.

BAB 5 PENUTUP

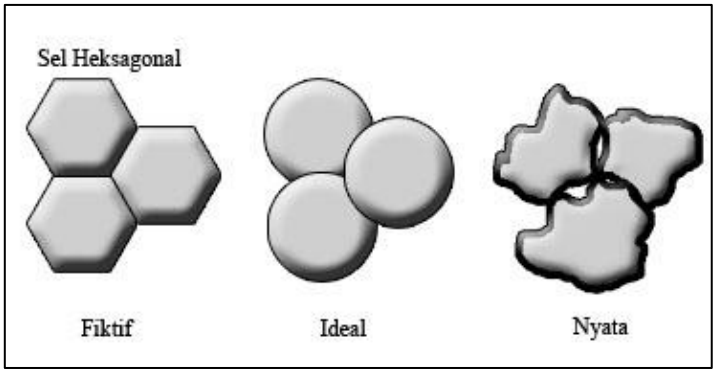
Merupakan bab yang berisi tentang kesimpulan dan saran.

BAB 2. TINJAUAN PUSTAKA

Bab tinjauan pustaka ini, diuraikan tentang teori yang mendukung perancangan perangkat lunak berbasis android untuk *drive test*. Beberapa teori yang ada di dalam sub bab telah dibatasi agar pembahasan tidak terlalu melebar dan supaya terfokus pada teori yang dibutuhkan untuk penyelesaian permasalahan pada tugas akhir ini. *Drive test* merupakan bagian dari optimasi jaringan untuk sistem komunikasi bergerak seluler. Maka dari itu berikut ini di uraikan teori-teori yang menjelaskan subsistem pada telekomunikasi.

2.1 Sistem Komunikasi Bergerak

Dalam sistem radio seluler terdapat empat bentuk sel yaitu lingkaran, segitiga sama sisi, bujur sangkar dan segi enam beraturan (heksagonal). Bentuk sel yang paling cocok untuk sistem radio seluler adalah berbentuk heksagonal karena dengan radius sel yang sedikit untuk mencakup wilayah pelayanan dibandingkan dengan bentuk sel segitiga atau bujur sangkar (Wibisono, 2007). Bentuk sel yang sesungguhnya tidak beraturan dan bergantung pada kontur permukaan daerah seperti terlihat pada gambar 2.1.



Gambar 2.1 Kondisi sel heksagonal dan bentuk cakupannya

Beberapa faktor yang paling mempengaruhi ukuran sel didalam suatu daerah layanan yaitu : kepadatan trafik telepon, kekuatan pemancar, sensitivitas penerima dari *Node B* maupun dari UE, tinggi antena dan keadaan topografinya.

Bentuk sel heksagonal sering digunakan dalam perencanaan karena dapat mempermudah dalam menganalisa. Luas daerah yang diliputi oleh sebuah sel heksagonal adalah sebesar :

$$L = \frac{3}{2} \times \sqrt{3} \times R^2 \quad \text{..... (2.1)}$$

dimana : L = Luas daerah sebuah sel hesagonal (Km^2)

R = Jari – jari sebuah sel (Km)

2.1.1 Mekanisme Propagasi Gelombang

Tiga mekanisme propagasi elektromagnetik dalam sistem komunikasi *wireless* adalah *reflection* (pantulan), *diffraction* (difraksi) dan *scattering* (hamburan). Pantulan terjadi ketika perambatan gelombang mengenai objek yang ukurannya jauh lebih besar dari panjang gelombang yang di pancarkan. Difraksi terjadi pada saat gelombang yang dipancarkan dibelokkan oleh benda dengan permukaan yang memiliki sisi yang tajam. Hamburan terjadi ketika medium yang dilewati gelombang terjadi dari benda-benda yang ukurannya lebih kecil dibandingkan dengan panjang gelombangnya. Daya yang diterima oleh *receiver* sejarak *d* dari *transmitter* dinyatakan oleh :

$$P_r(d) = P_t G_t G_r \frac{h_t^2 h_r^2}{d^4} \quad \text{.....(2.2)}$$

Dimana P_r adalah daya yang diterima, G_t dan G_r masing-masing adalah gain antena pemancar dan penerima, h_t dan h_r masing-masing adalah tinggi antena pemancar dan penerima, dan *d* adalah jarak antara pemancar dengan penerima.

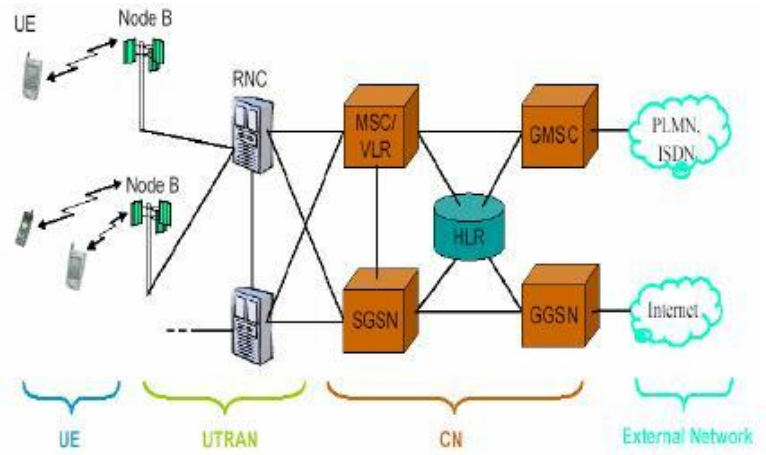
.

2.1.2 Konsep Dasar Jaringan 3G WCDMA

Teknologi komunikasi bergerak generasi ketiga (3G) yaitu *Universal Mobile Telecommunication System* (UMTS). UMTS merupakan suatu evolusi dari GSM, dimana *interface* radionya adalah WCDMA (*Wideband Code Division Multiple Access)* dimana teknologi ini memungkinkan kecepatan data mencapai 384 Kbps. Teknologi WCDMA sangat berbeda dengan teknologi jaringan GSM. Pada jaringan 3G dibutuhkan kualitas suara yang baik, data rate yang semakin tinggi (mencapai 2Mbps dengan menggunakan *release* 99, dan mencapai 10Mbps dengan menggunakan HSDPA) oleh sebab itu *bandwidth* sebesar 5 Mhz dibutuhkan pada sistem WCDMA. Posibilitas setiap *user* untuk mendapatkan *bandwidth* yang bervariasi sesuai permintaan layanan pengguna adalah salah satu fitur keunggulan jaringan UMTS. Teknik diversitas digunakan untuk menigkatkan kapasitas pengguna *downlink*, aktifitas *frequency planning* yang rumit pada jaringan GSM tidak perlu dilakukan. *Packet data scheduling* tergantung pada kapasitas jaringan sehingga lebih efisien dibandingkan jaringan GSM yang bergantung pada kapasitas *timeslot.*

2.1.3 Arsitektur Jaringan 3G

Teknologi telekomunikasi *wireless* generasi ketiga (3G) yaitu *Universal Mobile Telecommunication System* (UMTS). *Universal Mobile Telecommunication System* merupakan suatu evolusi dari GSM, dimana *interface* radionya adalah WCDMA, serta mampu melayani transmisi data dengan kecepatan yang lebih tinggi. Arsitektur jaringan UMTS terlihat pada Gambar .



Gambar 2.2 Arsitektur jaringan 3G WCDMA.

[Sumber :http://digilib.ittelkom.ac.id, 2008]

Dari gambar diatas terlihat bahwa arsitektur jaringan UMTS terdiri dari perangkat-perangkat yang saling mendukung, yaitu *User Equipment* (UE), *UMTS Terresterial Radio Access Network* (UTRAN) dan *Core Network* (CN).

1. UE *(User Equipment)*

User Equipment merupakan perangkat yang digunakan oleh pelanggan untuk dapat memperoleh layanan komunikasi bergerak. UE dilengkapi dengan *smart card* yang dikenal dengan nama USIM *(UMTS Subscriber Identity Module)* yang berisi nomor identitas pelanggan dan juga algoritma *security* untuk keamanan seperti *authentication algorithm* dan algoritma enkripsi. Selain terdapat USIM, UE juga dilengkapi dengan ME *(Mobile Equipment)* yang berfungsi sebagai terminal radio yang digunakan untuk komunikasi lewat radio.

2. UTRAN *(UMTS Terresterial Radio Access Network)*

Jaringan akses radio menyediakan koneksi antara terminal *mobile* dan *Core Network*. Dalam UMTS jaringan akses dinamakan UTRAN (*Access Universal Radio electric Terrestrial*). UTRA mode UTRAN terdiri dari satu atau lebih Jaringan Sub-

Sistem Radio (RNS). Sebuah RNS merupakan suatu sub-jaringan dalam UTRAN dan terdiri dari *Radio Network Controller* (RNC) dan satu atau lebih *Node B*. RNS dihubungkan antar RNC melalui suatu Iur *Interface* dan *Node B* dihubungkan dengan satu Iub *Interface*.

Di dalam UTRAN terdapat beberapa elemen jaringan yang baru dibandingkan dengan teknologi 2G yang ada saat ini, di antaranya adalah *Node B* dan RNC *(Radio Network Controller)*.

a. RNC *(Radio Network Controller)*

RNC bertanggung jawab mengontrol *radio resources* pada UTRAN yang membawahi beberapa *Node B*, menghubungkan CN *(Core Network)* dengan *user*, dan merupakan tempat berakhirnya protokol RRC *(Radio Resource Control)* yang mendefinisikan pesan dan prosedur antara *mobile user* dengan UTRAN.

b. *Node B*

Node B sama dengan *Base Station* di dalam jaringan GSM. *Node B* merupakan perangkat pemancar dan penerima yang memberikan pelayanan radio kepada UE. Fungsi utama *Node B* adalah melakukan proses pada *layer* 1 antara lain : *channel coding, interleaving*, *spreading*, *de-spreading,* modulasi, demodulasi dan lain-lain. *Node B* juga melakukan beberapa operasi RRM *(Radio Resouce Management),* seperti *handover* dan *power control.*

3. CN *(Core Network)*

Jaringan Lokal (*Core Network*) menggabungkan fungsi kecerdasan dan transport. *Core Network* ini mendukung pensinyalan dan transport informasi dari trafik, termasuk peringanan beban trafik. Fungsi-fungsi kecerdasan yang terdapat langsung seperti logika dan dengan adanya keuntungan fasilitas kendali dari layanan melalui antarmuka yang terdefinisi jelas yang juga pengaturan mobilitas. Dengan

melewati inti jaringan, UMTS juga dihubungkan dengan jaringan telekomunikasi lain, jadi sangat memungkinkan tidak hanya antara pengguna UMTS *mobile*, tetapi juga dengan jaringan yang lain.

a. MSC *(Mobile Switching Center)*

MSC didesain sebagai *switching* untuk layanan berbasis *circuit switch* seperti *video, video call.*

b. VLR *(Visitor Location Register)*

VLR merupakan database yang berisi informasi sementara mengenai pelanggan terutama mengenai lokasi dari pelanggan pada cakupan area jaringan.

c. HLR *(Home Location Register)*

HLR merupakan database yang berisi data-data pelanggan yang tetap. Data-data tersebut antara lain berisi layanan pelanggan, *service* tambahan serta informasi mengenai lokasi pelanggan yang paling akhir (*Update Location*).

d. SGSN *(Serving GPRS Support Node)*

SGSN merupakan gerbang penghubung jaringan BSS/BTS ke jaringan GPRS. Fungsi SGSN adalah sebagai berikut :

1. Mengantarkan paket data ke MS.
2. Update pelanggan ke HLR.
3. Registrasi pelanggan baru.

e. GGSN *(Gateway GPRS Support Node)*

GGSN berfungsi sebagai gerbang penghubung dari jaringan GPRS ke jaringan paket data standard (PDN). GGSN berfungsi dalam menyediakan fasilitas *internet working* dengan *eksternal packet-switch network* dan dihubungkan dengan

SGSN *via Internet Protokol* (IP). GGSN akan berperan antarmuka logik bagi PDN, dimana GGSN akan memancarkan dan menerima paket data dari SGSN atau PDN. Selain itu juga terdapat beberapa *interface* baru, seperti : Uu, Iu, Iub, Iur. Antara UE dan UTRAN terdapat *interface* Uu. Di dalam UTRAN terdapat *interface* Iub yang menghubungkan *Node B* dan RNC, *Interface* Iur yang menghubungkan antar RNC, sedangkan UTRAN dan CN dihubungkan oleh *interface* Iu.

Protokol pada *interface* Uu dan Iu dibagi menjadi dua sesuai fungsinya, yaitu bagian *control plane* dan *user plane* . Bagian *user plane* merupakan protokol yang mengimplementasikan layanan *Radio Access Bearer* (RAB), misalnya membawa data *user* melalui *Access Stratum* (AS). Sedangkan *control plane* berfungsi mengontrol RAB dan koneksi antara *mobile user* dengan jaringan dari aspek jenis layanan yang diminta, pengontrolan sumber daya transmisi, *handover*, mekanisme transfer *Non Access Stratum* (NAS) seperti *Mobility Management* (MM), *Connection Managemen*t (CM), *Session Management* (SM) dan lain-lain.

2.2 Optimasi

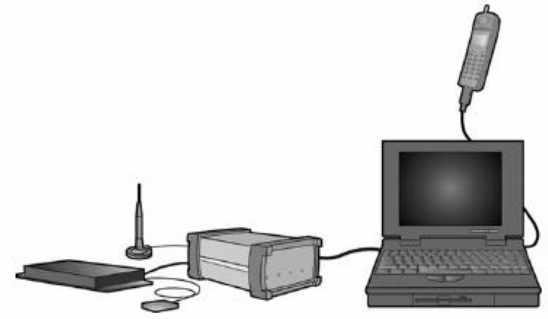
Optimasi merupakan langkah penting dalam siklus hidup suatu jaringan. *Drive test* merupakan langkah awal proses, dengan tujuan untuk mengumpulkan data pengukuran yang berkaitan dengan lokasi *user*. Setelah data terkumpul sepanjang luas cakupan RF yang diinginkan, maka data ini akan diproses pada suatu perangkat lunak tertentu. Setelah masalah, penyebab dan solusi dapat diidentifikasi, langkah selanjutnya adalah melakukan pemecahan masalah tersebut.

2.2.1 *Drive Test*

Drive Test adalah proses pengukuran sistem komunikasi bergerak pada sisi gelombang radio di udara yaitu dari arah pemancar/BTS ke MS/*handphone* atau sebaliknya, dengan menggunakan *handphone* yang didesain secara khusus untuk

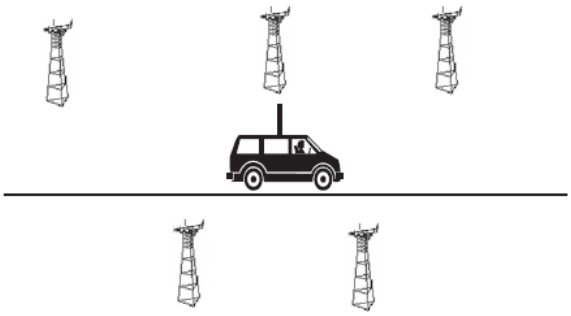
pengukuran. *Drive test* memungkinkan *operator* untuk melakukan optimasi yang terus berjalan. Umumnya, *drive test* dilakukan dengan menghubungkan MS ke laptop. Perangkat *drive test* menggunakan MS untuk mensimulasikan masalah yang dialami pelanggan ketika akan atau saat melakukan panggilan. Sebagai contoh, jika panggilan pelanggan terputus ketika beroperasi di dalam obyek bergerak pada suatu lokasi tertentu, maka perangkat *drive test* harus mampu mensimulasikan masalah ini.

Contoh lain masalah yang dialami pelanggan adalah panggilan yang diblokir (kegagalan mendapatkan akses), kualitas suara yang buruk, dan cakupan area pelayanan yang kurang. Sistem *drive test* melakukan pengukuran, menyimpan data di komputer, dan menampilkan data menurut waktu dan tempat. Beberapa tipe *system drive test* yang tersedia berbasis MS, berbasis *receiver* dan kombinasi keduanya. Gambar 2.3 menunjukkan *system drive test* kombinasi antara MS dan *receiver*. Sistem *drive test* diterapkan dalam kendaraan dan dikemudikan sepanjang area cakupan *operator*. Gambar 2.4 menunjukkan proses *drive test* dalam mobil.



Gambar 2.3 Konfigurasi *drive test MS-receiver*

[Sumber :http://1.bp.blogspot.com/WQYYJvtMZX0/TqfnBwZEXTI/AAAAAAAAAEk/T3M86X0J6Xo/s1600/Konfigurasi%2BMS.jpg]



Gambar 2.4 Proses *drive test* dalam mobil

[Sumber :http://3.bp.blogspot.com/-cRqzmBGv-gY/TqfnTJ8NGlI/AAAAAAAAAEw/DLwGLvP3jT0/s1600/DT%2Bmobil.jpg]

2.2.2 Parameter Analisis *Drive Test*

Menurut R Bram Aditya (2011:3) terdapat beberapa parameter yang dijadikan referensi umum untuk analisis kualitas panggilan suara, yaitu CPICH RSCP (*Common Pilot Channel Received Signal Code Power)*, CPICH Ec/No (*Common Pilot Channel Ec/*No), SC (*Scrambling Code*, *Call Setup Success Ratio* (CSSR), *Drop Call Ratio* (DCR).

1. CPICH RSCP (*Common Pilot Channel Received Signal Code Power*)

CPICH RSCP adalah kuat sinyal penerimaan yang menyatakan besarnya daya pada satu kode yang diterima oleh UE (*User Equipment*) yang merupakan salah satu parameter yang menentukan nilai Ec/No. Nilai CPICH RSCP merupakan suatu nilai yang menunjukkan level kekuatan sinyal.

Tidak ada standar yang ditetapkan untuk nilai CPICH RSCP. Setiap operator memiliki ambang yang berbeda-beda. Nilai CPICH RSCP yang digunakan pada Tugas Akhir ini dapat dilihat pada Tabel 2.1.

Standar warna	Range nilai	Keterangan
	-82 sampai 0	Sangat baik
	-88 sampai -82	Baik
	-92 sampai -88	Sedang
	-104 sampai -92	Buruk
	-130 sampai -104	Sangat buruk

Tabel 2. 1 Range nilai RSCP (dBm)

(Sumber : *Drive test report* PT. CM Tech, 2013)

2. CPICH Ec/No (*Common Pilot Channel Ec/No*)

CPICH Ec/No adalah rasio perbandingan antara energi yang dihasilkan dari sinyal pilot dengan total energi yang diterima. Ec/No juga menunjukkan level daya minimum (*threshold*) dimana MS masih bisa melakukan suatu panggilan. Rasio perbandingan antara energi yang dihasilkan dari setiap pilot dengan total energi yang diterima diberikan oleh persamaan berikut :

CPICH RSCP = RSSI + CPICH Ec/No ……(2.3)

di mana :

CPICH Ec/No= rasio perbandingan antara energi yang dihasilkan dari sinyal pilot dengan total energi yang diterima (dB)

CPICH RSCP = *Received Signal Code Power*(dBm)

RSSI = *Receive Signal Strength Interference*(dBm)

Tidak ada standar yang ditetapkan untuk nilai CPICH Ec/No. Setiap operator memiliki ambang yang berbeda-beda. Nilai CPICH Ec/No yang digunakan pada Tugas Akhir ini dapat dilihat pada Tabel 2.2.

Standar warna	Range nilai	Keterangan
	-9 sampai 0	Sangat baik
	-12 sampai -9	Baik
	-15 sampai -12	Buruk
	-25 sampai -15	Sangat buruk

Tabel 2.2 Range nilai Ec/No (dBm)

(Sumber : *Drive test report* PT. CM Tech, 2013)

3. *Scrambling Code* (SC)

Scrambling Code berguna untuk membedakan UE yang satu dengan UE yang lain disisi *uplink* dan juga untuk membedakan *node B* yang satu dengan *node B* yang lainnya disisi *downlink*. Sehingga SC memberikan informasi sektor dari site yang sedang *serving*.

4. *Call Setup Success Ratio* (CSSR)

CSSR adalah prosentase tingkat keberhasilan melakukan setup panggilan sehingga diperoleh kanal yang dipergunakan pada saat awal *signaling*. Pada perhitungan CSSR menggunakan rumusan sebagai berikut :

$$CSSR = 100x\left[\frac{\Sigma callsetup}{\Sigma callattempt}\right] \ldots\ldots\ldots\ldots (2.4)$$

5. *Call Drop Ratio*

Call Drop Ratio adalah prosentase banyaknya panggilan yang jatuh atau putus setelah kanal pembicaraan digunakan. Pada perhitungan *call drop ratio* ini digunakan menggunakan rumus sebagai berikut :

$$CallDropRatio(\%) = 100x\left[\frac{\Sigma callDropped}{\Sigma callSetup}\right] \ldots\ldots\ldots\ldots (2.5)$$

Call drop dapat disebabkan beberapa hal, antara lain :

- Rugi-rugi frekuensi radio
- *Co Chanal interferensi* dan *Adjecent interferensi*
- Kegagalan prose *handover*

2.3 GPS

GPS (*Global Positioning System*) adalah sistem satelit navigasi dan penentuan posisi yang dimiliki dan dikelola oleh Amerika Serikat. Sistem ini didesain untuk memberikan posisi dan kecepatan tiga-dimensi serta informasi mengenai waktu, secara kontinyu di seluruh dunia tanpa bergantung waktu dan cuaca, bagi banyak orang secara simultan. GPS dapat memberikan informasi posisi dengan ketelitian bervariasi dari beberapa millimeter (orde nol) sampai dengan puluhan meter.

Beberapa kemampuan GPS antara lain dapat memberikan informasi tentang posisi, kecepatan, dan waktu secara cepat, akurat, murah, dimana saja di bumi ini tanpa tergantung cuaca. Ketelitian dari GPS dapat mencapai beberapa milimeter untuk ketelitian posisinya, beberapa centimeter per second (cm/s) untuk ketelitian kecepatannya dan beberapa nanodetik untuk ketelitian waktunya. Ketelitian posisi yang diperoleh akan tergantung pada beberapa faktor yaitu metode penentuan posisi, geometri satelit, tingkat ketelitian data, dan metode pengolahan datanya.

GPS menggunakan perangkat satelit sejumlah 24 perangkat (jumlah pastinya berjumlah 27 perangkat satelit dengan 3 perangkat sebagai cadangan saat perangkat satelit yang lain tidak dapat berfungsi dengan baik) yang mengirimkan sinya mikro ke bumi. Kemudian sinyal mikro tersebut ditangkap dan diterima oleh perangkat GPS yang ada di bumi dan data yang ditangkap tersebut diolah untuk mendapatkan informasi lokasi, kecepatan, arah dan waktu.

Cara kerja GPS pada ponsel memiliki hampir sama dengan cara kerja ponsel terhubung dengan operator yang digunakan. Sebuah ponsel memiliki kemampuan

berkomunikasi 2 arah dengan BTS (*Base Transceiver Station)* atau *node B* pada WCDMA. Saat ponsel berpindah dari satu tempat ke tempat lain, maka secara tidak sadar ponsel menangkap sinya dari BTS yang berbeda namun masih dalam 1 operator. Sehingga operator telekomunikasi mengetahui keberadaan ponsel secara presisi dari letak menara BTS yang sinyanya ditangkap oleh ponsel.

2.4 Java dan Android

Java merupakan perangkat lunak produksi *Sun Microsystem Inc*. untuk pemrograman beberapa tujuan (*multipurpose*), dapat berjalan di beberapa sistem operasi (*multiplatform*), mudah dipelajari dan *powerful*. Aplikasi-aplikasi yang dapat dibuat dengan java, meliputi *web programming, Desktop programming,*dan *mobile programming.*

2.4.1 Mengenal Java 2

Perangkat lunak Java sintaknya (tulisannya) mirip dengan C, karena bahasa Java dibuat memakai bahasa pemrograman C, tetapi bahasa Java menyempurnakan kekurangan C. Pertama rilis Java disebut JDK (Java *Development Kit*), hingga JDK Versi 2 atau dikenal dengan Java 2, dibagi menjadi 3 edisi, yaitu J2SE (Java 2 *Standart Edition*), J2EE (Java 2 *Entrerprise Edition*), dan J2ME (Java 2 *Micro Edition*).

J2SE merupakan edisi atau teknologi untuk pemrograman desktop atau aplikasi layar (*console*). J2SE juga merupakan perangkat lunak dasar yang harus diinstal sebelum memakai J2EE dan J2ME. J2EE merupakan edisi atau teknologi untuk pemrograman *enterprise*, seperti pemrograman *database*, JSP, Beans, dan lain-lainnya. J2ME merupakan edisi atau teknologi untuk pemrograman *mobile* atau *handphone* dan peralatan kecil (*smal ldevice*).

Setelah menginstal Java, di dalam sistem komputer terdapat JVM (Java *Virtual Machine*). Di dalam JVM terdapat JRE (Java *Runtime Enviroment*). Program

yang diketik dengan bahasa java memiliki ekstensi .java, akan menghasilkan *file .class* jika dikompilasi. *File* kelas (*.class*) dapat dijalankan dengan memanfaatkan JRE.

2.4.2 Mengenal Android

Android merupakan sebuah sistem operasi perangkat mobile berbasis linux yang mencakup sistem operasi, *middleware*, dan aplikasi. Beberapa pengertian lain dari android, yaitu :

a. Merupakan platform terbuka (*opensource)* bagi para pengembang atau programmer untuk membuat aplikasi.
b. Merupakan sistem operasi yang dibeli Google Inc. dari Android Inc.
c. Bukan bahasa pemrograman, akan tetapi hanya menyediakan lingkungan hidup atau *runtime enviroment* yang disebut DVM (*Dalvik Virtual Machine*) yang telah dioptimasi untuk *device* atau alat dengan sistem memori yang kecil.

Untuk mengembangan Android, dibentuk OHA (*Open Handset Aliance*), konsorium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

Pada tanggal 5 November 2007, Android rilis pertama kali. Android bersama OHA menyatakan mendukung pengembangan *opensource* pada perangkat *mobile*. Sekitar bulan September 2007, Google mengenalkan Nexus One, salah satu jenis *smartphone* yang menggunakan android sebagai sistem operasinya. Telepon seluler ini diproduksi oleh HTC Corp dan tersedia di pasaran tanggal 5 Januari 2008.

Pada masa sekarang ini banyak vendor-vendor yang menggunakan sistem operasi Android untuk *smartphone* mereka, seperti HTC, Motorola, Samsung, LG, HKC, Huawei, Archos, Webstation Camangi, Dell, Nexus, SciPhone, WayteQ, Sony Ericsson, Acer, Philips, T-Mobile, Nexian, IMO, Asus, dan masih banyak lagi vendor-vendor lain.

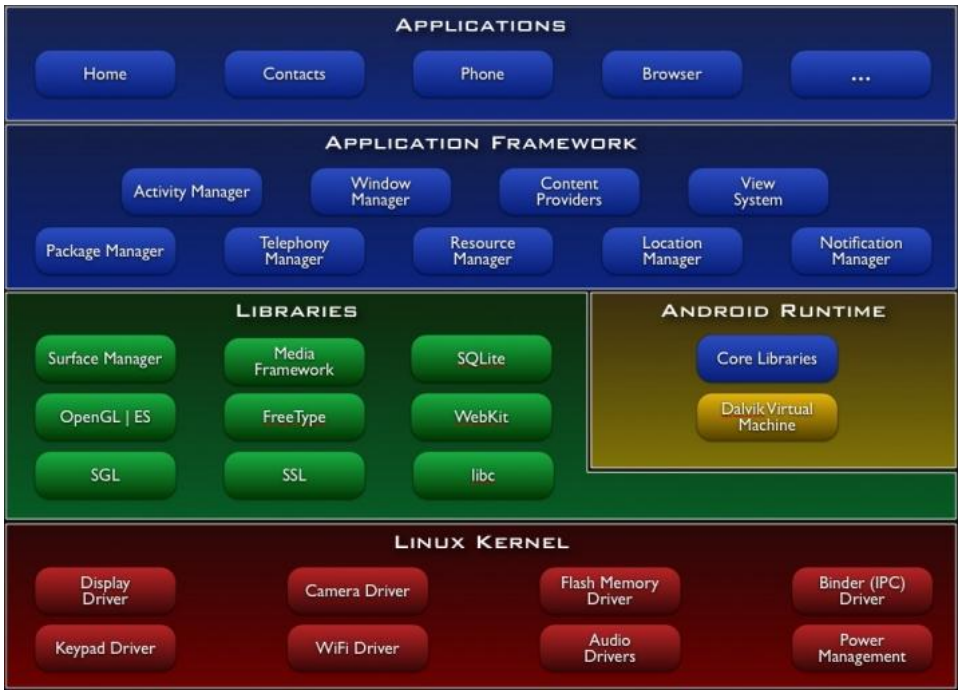
Pada saat ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau Google *Mail Service* (GMS). Dan

kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung Google, atau dikenal *Open Handset Distribution* (OHD).

Android menggunakan bahasa pemrograman Java. Android hanya menyediakan lingkungan *runtime* atau sebagi *interpreter*. Dimana kode sumber yang telah kita *compile* dengan *compiler* Java akan dioptimasi oleh *Delvik*. Sebuah *Virtual machine* yang memang dibuat khusus untuk menjalankan kode-kode program yang dibuat dengan bahasa pemrograman Java. Yang tentunya berbentuk sebuah *class*. Kemudian *dex tools* (merupakan bagian dari DVM) mengubah Java *class* yang telah di-*compile* oleh Java *Compiler* ke lingkungan *native* yang berbentuk *.dex format (*Dalvik executable*), yang teroptimasi untuk lingkungan perangkat keras dengan komputasi yang rendah.

2.4.3 Arsitektur Android

Secara garis besar, arsitektur android terdiri atas *aplications* dan *widget*, *aplication framework*, *libraries*, android *runtime* dan *linux kernel* (Naufal, 2012). Untuk lebih jelasnya lihat gambar 2.5 dibawah ini.



Gambar 2.5 Arsitektur android

(Sumber : tips-droid.blogspot.com)

- ***Layer Applications* dan *Widget*** : inilah *layer* pertama pada *OS* Android, biasa dinamakan *layer Applications* dan *widget*. *Layer* ini merupakan *layer* yang berhubungan dengan aplikasi-aplikasi inti yang berjalan pada android. Seperti klien email, program SMS, kalender, browser, peta, kontak, dan lain-lain. Apabila seorang *programmer* membuat aplikasi, maka aplikasi itu ada di *layer* ini.
- ***Layer Applications Framework*:** merupakan *layer* dimana para pembuat aplikasi menggunakan komponen-komponen yang ada di sini untuk membuat aplikasi mereka. Beberapa contoh komponen yang termasuk di dalam *Applications framework* adalah sebagai berikut:

a. *Views*
b. *Content provider*
c. *Resource manager*

d. *Notification manager*

e. *Activity manager*

- ***Layer Libraries*:** merupakan *layer* tempat fitur-fitur android berada. Pada umumnya *libraries* diakses untuk menjalankan aplikasi. Beberapa *library* yang terdapat pada android diantaranya adalah *libraries* media untuk memutar media video atau audio, *libraries* untuk menjalankan tampilan, *libraries Graphic*, *libraries* SQLite untuk dukungan *database*, dan masih banyak *library* lainnya.
- **Android *RunTime***: merupakan *layer* yang membuat aplikasi android bisa dijalankan. Android *runtime* dibagi menjadi dua bagian yaitu:

a. *Core Libraries* : berfungsi untuk menerjemahkan bahasa Java/C

b. *Dalvik Virtual Machine* : sebuah mesin virtual berbasis *register* yang dioptimalkan untuk menjalankan fungsi-fungsi pada android secara efisien.

- ***Linux Kernel*:** merupakan *layer* tempat keberadaan inti dari *operating system* android. *Layer* ini berisi *file-file* sistem yang mengatur sistem *processing*, *memory*, *resource*, *drivers*, dan sistem android lainnya.

2.4.4 Keunggulan Android

Beberapa keunggulan Android dengan *platform* lain yaitu :

1. Keterbukaan - Android menyediakan akses ke fungsi dasar perangkat *mobile* menggunakan standar panggilan ke API.
2. Penghancuran perbatasan – anda dapat menggabungkan informasi dari Internet ke dalam telepon, seperti informasi kontak, atau data pada lokasi geografis.
3. Cepat dan mudah perkembangannya - dalam SDK memiliki semua yang anda butuhkan untuk membuat dan menjalankan aplikasi Android, termasuk simulator dan alat *debugging*.

BAB 3. METODOLOGI PENELITIAN

Bab metode penelitian menjelaskan beberapa hal pokok yaitu studi pustaka yang digunakan, parameter atau obyek penelitian, cara pengamatan dan pengujian, tempat dan waktu penelitian, langkah-langkah dalam pengumpulan data dan manajemen penelitian di lapangan, pengolahan data serta analisis data yang dipakai. Semuanya dijelaskan secara cermat dan jelas.

3.1 Studi Pustaka

Studi Pustaka dilakukan untuk mempelajari :

1. Pemrograman Android dengan Java

   Studi pemrograman android dengan java bertujuan untuk mengetahui dasar-dasar perancangan aplikasi berbasis android.
2. *Map, Marker,* dan GPS

   Studi ini bertujuan untuk menampilkan map, menentukan lokasi dari GPS dan memberi tanda lokasi dengan sebuah marker.
3. *Drive test* pada jaringan 3G (WCDMA)

   Studi drive test pada jaringan 3G (WCDMA) digunakan untuk mempelajari parameter yang harus diamati ketika melakukan pengambilan data serta cara menganalisa data yang telah diperoleh dari proses *drive test.*

3.2 Waktu dan Tempat Penelitian

3.2.1 Tempat Penelitian

Perancangan perangkat lunak ini akan dilakukan di Jurusan Elektro, Fakultas Teknik, Universitas Jember. Untuk analisis kinerja dari perangkat lunak ini akan dilakukan di PT Cahaya Mitratama *Technology* Denpasar, Bali, dengan tempat pengambilan data *drive test* di daerah *cluster* Uluwatu.

3.2.2 Waktu Penelitian

Waktu pelaksanaan penelitian mulai dari studi pustaka hingga penyusunan laporan dilakukan selama satu semester dengan rincian sebagai berikut:

NO	JENIS KEGIATAN	BULAN				
		I	II	III	IV	V
1	Studi Pustaka					
2	Perancangan					
3	Pengambilan Data dan Pengujian					
4	Pembahasan					
5	Penyusunan Laporan					

Table 3.1 Jadwal kegiatan penelitian

3.3 Alat dan Bahan

3.3.1 Alat

1. Satu buah Komputer: digunakan untuk meng-*instal software* pemrograman android dan penyusunan laporan tugas akhir.
2. *Smartphone* android : digunakan untuk pengujian dari perangkat lunak.

3.3.2 Bahan

1. Android Studio
2. MapInfo Profesional 11

3.4 Parameter Penelitian

Ada beberapa parameter yang akan digunakan dalam penelitian perancangan perangkat lunak *drive test* ini. Parameter tersebut antara lain:

1. Kualitas sinyal yang diterima MS*;*
2. Kuat sinyal yang diterima MS;
3. *Call Setup Success Ratio* (CSSR);
4. *Call Drop Ratio*(CDR)*;*
5. *Scrambling Code* (SC);

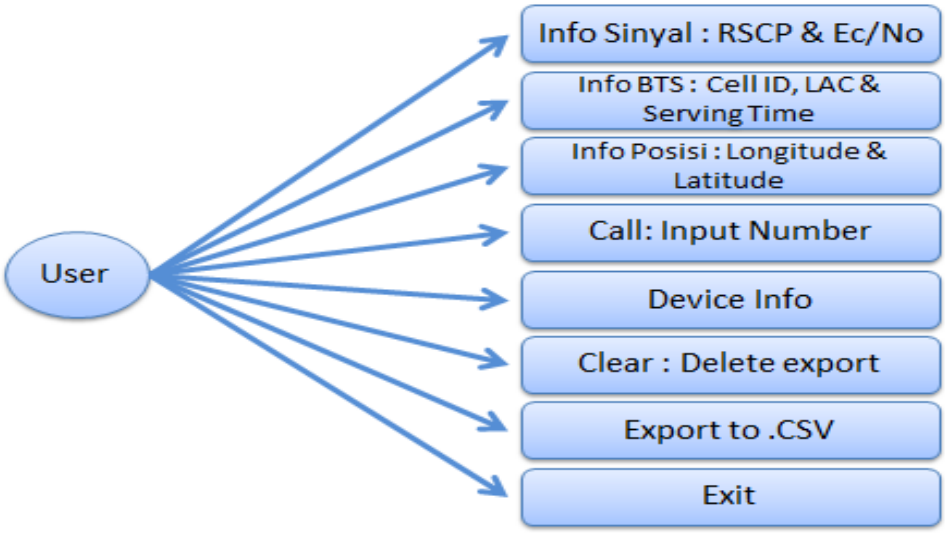
3.5 Tahap Perancangan

Tahap perancangan yang pertama dilakukan adalah melengkapi semua alat dan bahan yang diperlukan. sebelum memulai pemrograman, terlebih dahulu melakukan instalasi *Android Studio* sebagai editor pemrograman java. Selanjutnya instalasi Android SDK yang merupakan alat untuk membuat aplikasi *platform* android.

Setelah mengetahui parameter *drive test* analisis kualitas *voice call*, selanjutnya memulai merancang *user interface* dan memulai perancangan perangkat lunak secara keseluruhan.

3.5.1 *Use Case* Diagram

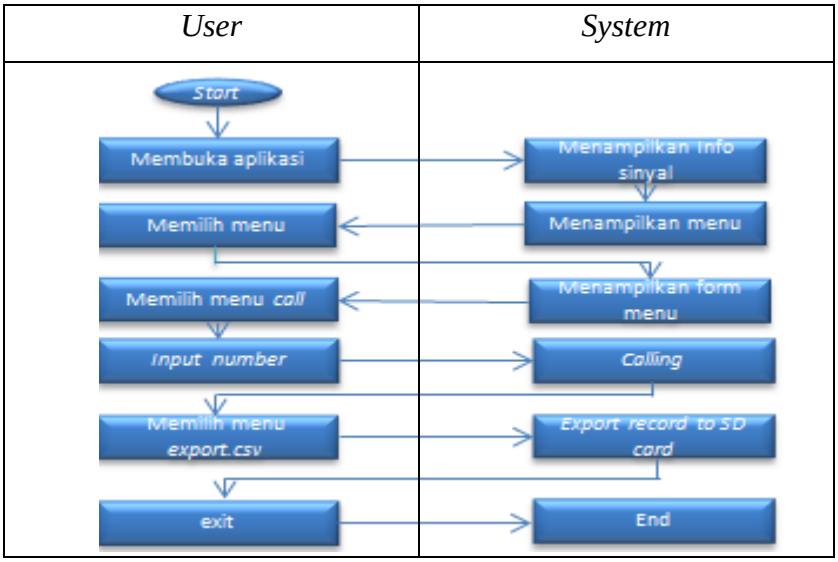
Use case merupakan gambaran skenario dari interaksi antara *user* dengan sistem. Sebuah diagram *use case* menggambarkan hubungan antara aktor (pengguna) dan kegiatan yang dapat dilakukannya terhadap aplikasi. Berikut ini adalah *use case* diagram yang memperlihatkan peranan aktor dalam interaksinya dengan sistem.



Gambar 3.1 *Use case diagram*

3.5.2 *Activity* Diagram

Jalur kerja dari aplikasi *drive test* akan dijelaskan pada *Activity Diagram* dibawah ini :



Gambar 3.2 *Activity Diagram*

3.5.3 User Interface

Perancangan tampilan aplikasi ini menjelaskan bagaimana suatu aplikasi tertampil pada layar dan tata letak berserta daftar menu. Berikut adalah rancangan untuk tampilan menu dari perangkat lunak ini :

Gambar 3.3 Desain *Main Menu*

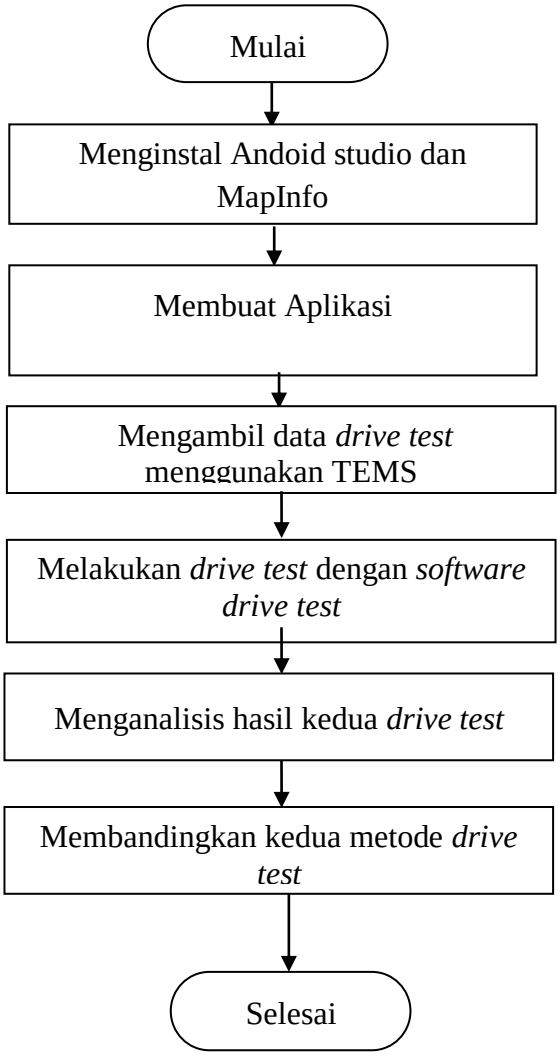
3.6 Pengambilan Data dan Analisis

Pengambilan data dilakukan dengan proses pengukuran beberapa parameter yang telah di sebutkan pada subbab 3.4 pengambilan data ini dibandingkan dengan pengambilan data *drive test* menggunakan metode yang sudah ada, hal ini bertujuan untuk mengetahui kualitas hasil rancangan. Berikut ini rincian untuk melakukan proses pengambilan data.

1. *Lock smartphone* pada jaringan 3G WCDMA.
2. Pengukuran dengan mengendarai sepeda motor pada kecepatan rata-rata 20km/jam.
3. Selanjutnya melakukan panggilan telfon selama 1 menit dan dilakukan berulang-ulang sampai lokasi *drive test* selesai.
4. Dan hasilnya di *export* ke csv.

5. Untuk analisis pengukuran kualitas sinyal dengan mengukur nilai RSCP.
6. Pengukuran kuat sinyal dengan mengukur nilai EcNo.
7. Melihat nilai Cell Id.
8. Mengukur jumlah panggilan yang sukses.

3.7 Diagram Alur Penelitian



Gambar 3.4 Diagram alur penelitian

BAB 4. PEMBAHASAN

Pada bab ini akan dijelaskan tentang perancangan aplikasi android *drive test* yang diberi nama Net Info, selanjutnya cara pengolahan data dari Net Info, analisis data berdasarkan tabel dan *plotting* gambar serta yang terakhir adalah membandingkan data yang diperoleh dari Net Info dengan *drive test* menggunakan TEMS.

4.1 Tahap Perancangan Sistem

4.1.1 Analisis Masalah

Tahap analisis masalah akan memberikan data dan opini atas permasalahan yang dibidik dan dicarikan solusinya. Pada tahap ini kita melibatkan beberapa *stakeholder* (pihak terkait) dalam menentukan tingkat kompleksitas masalah.

Pada penelitian ini bidikan analisis masalah difokuskan pada proses proses pengambilan data *drive test* yang masih terkendala dengan banyaknya alat yang dibutuhkan dalam proses *drive test.* Aplikasi yang dibutuhkan adalah memiliki fitur secara langsung mampu meminimalkan alat yang dibutuhkan dalam *drive test* serta mampu memberikan data dengan kevalidan yang terakui dan langkah operasional yang sederhana.

4.1.2 Analisis Kebutuhan Sistem

Tahap analisis sistem merupakan salah satu usaha mengidentifikasi kebutuhan dan spesifikasi sistem yang akan diciptakan. Di dalamnya akan dijabarkan apa saja proses yang dijalankan, serta output yang dihasilkan.

Berdasarkan hasil analisis masalah yang telah dijabarkan sebelumnya, solusi yang ditawarkan adalah pembuatan aplikasi *drive test* berbasis android yang dinamakan Net Info dengan spesifikasi sistem sebagai berikut :

1. Sistem dapat menampilkan parameter *drive test*.
2. Sistem dapat memfungsikan GPS *smartphone* dan menampilkan *longitude* dan *latitute*.
3. Sistem dapat melakukan panggilan dengan menggunakan fungsi *call*.
4. Sistem dapat menyimpan parameter *drive test* ke dalam bentuk tabel dengan format *CSV*.

4.1.3 Analisis Kebutuhan Fungsional dan Non Fungsional

4.1.3.1 Analisis Kebutuhan Fungsional

Mengidentifikasi fasilitas dan aktivitas apa saja yang seharusnya dikerjakan oleh sistem. Adapun kebutuhan fungsional dalam penelitian ini menghasilkan rincian sebagai berikut :

1. Fungsi melihat informasi parameter *drive test.*
2. Fungsi GPS untuk penentuan lokasi.
3. Fungsi *export to CSV.*
4. Fungsi *input number.*
5. Fungsi *calling.*

4.1.3.2 Analisis Kebutuhan Non Fungsional

Mengidentifikasi batasan dari fasilitas serta fungsi tambahan yang disediakan oleh sistem. Berikut rincian kebutuhan non fungsional dari dari sistem:

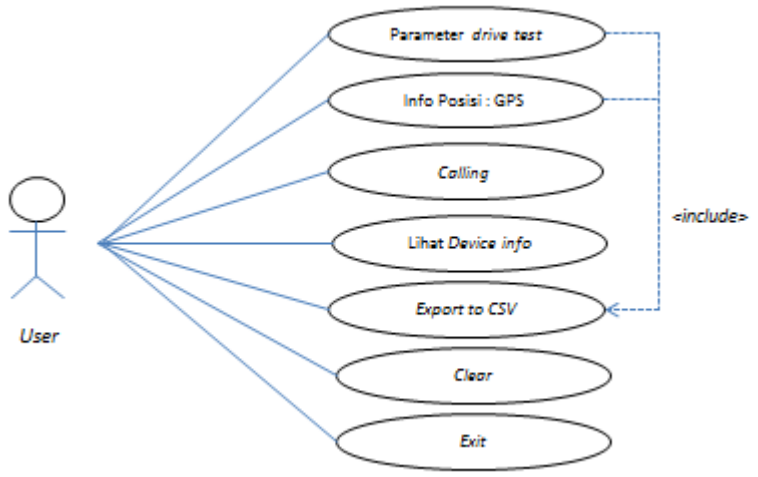
1. Fungsi *Service State* untuk mengetahui ketersediaan layanan dari operator.
2. *Connection State* untuk mengetahui status koneksi internet.
3. *Device Info* untuk mengetahui info dari *smartphone* seperti operator, tipe jaringan dan tipe *handphone.*

4.1.4 Perancangan Sistem

Perancangan sistem yang merupakan analisis kebutuhan fungsional akan mencakup *use case*, *activity* diagram, *class* diagram, desain tabel dan desain *interface.*

4.1.4.1 *Use Case*

Use Case berguna sebagai langkah awal untuk memodelkan interaksi tunggal antara pengguna dengan sistem. Berikut ini merupakan gambaran dari desain sistem aplikasi Net Info.

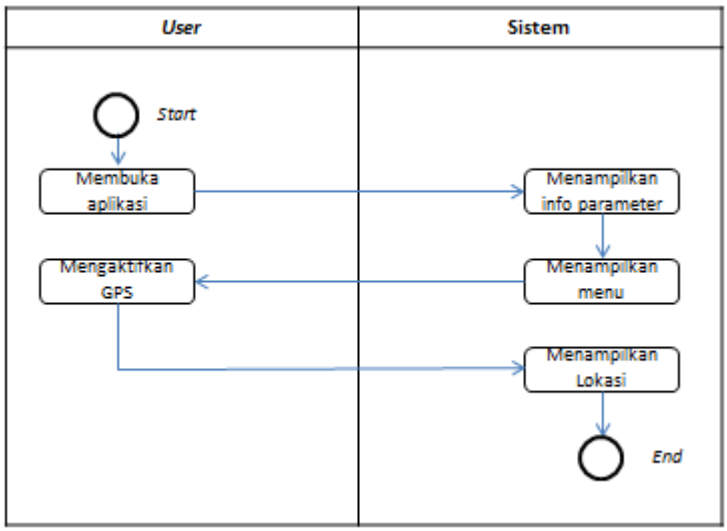


Gambar 4.1 *Use case* Net Info

4.1.4.2 *Activity* Diagram

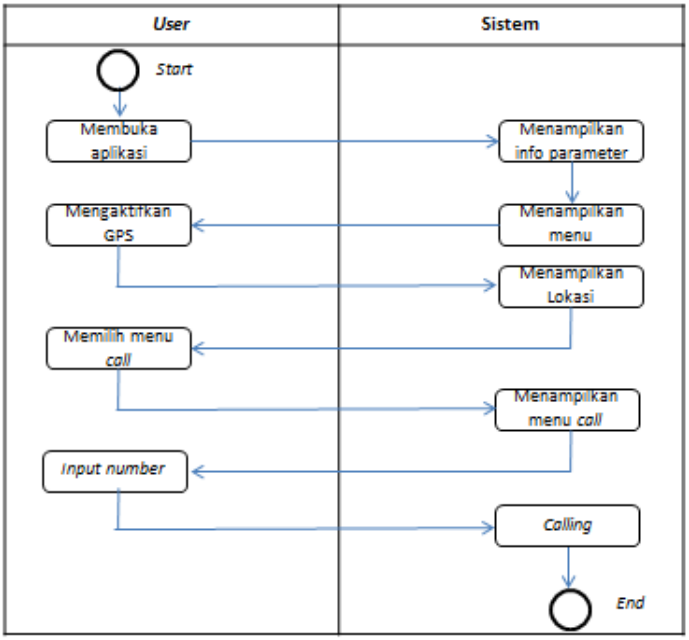
Activity diagram menggambarkan rangkaian aliran dari aktivitas *user* dengan aplikasi Net Info. Aktivitas ini akan dibagi menjadi beberapa kategori sesuai kegiatan yang *user* dapat lakukan.

1. *Activity* Diagram *Main Interface*



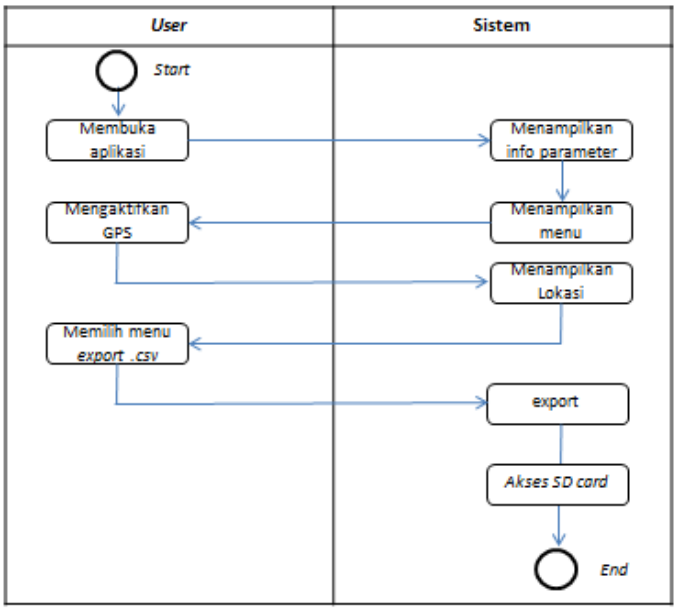
Gambar 4.2 *Activity* Diagram *Main Interface*

2. *Activity* Diagram *Calling*



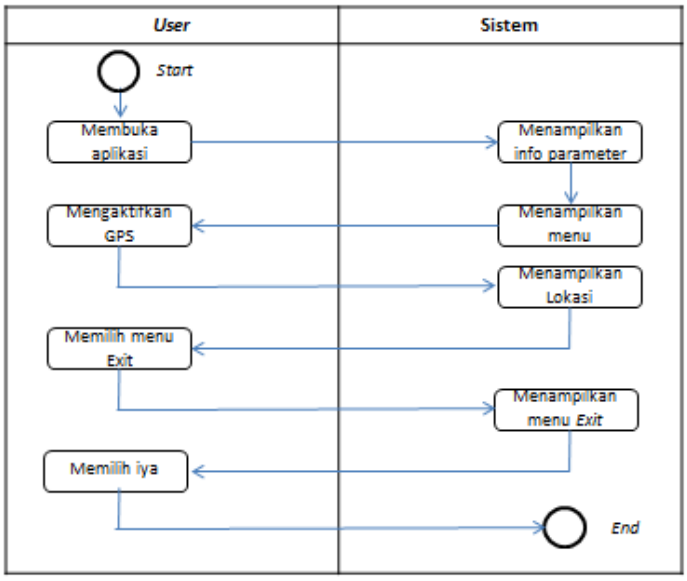
Gambar 4.3 *Activity* Diagram *Calling*

3. *Activity* Diagram *export to CSV*



Gambar 4.4 *Activity* Diagram *Export to CSV*

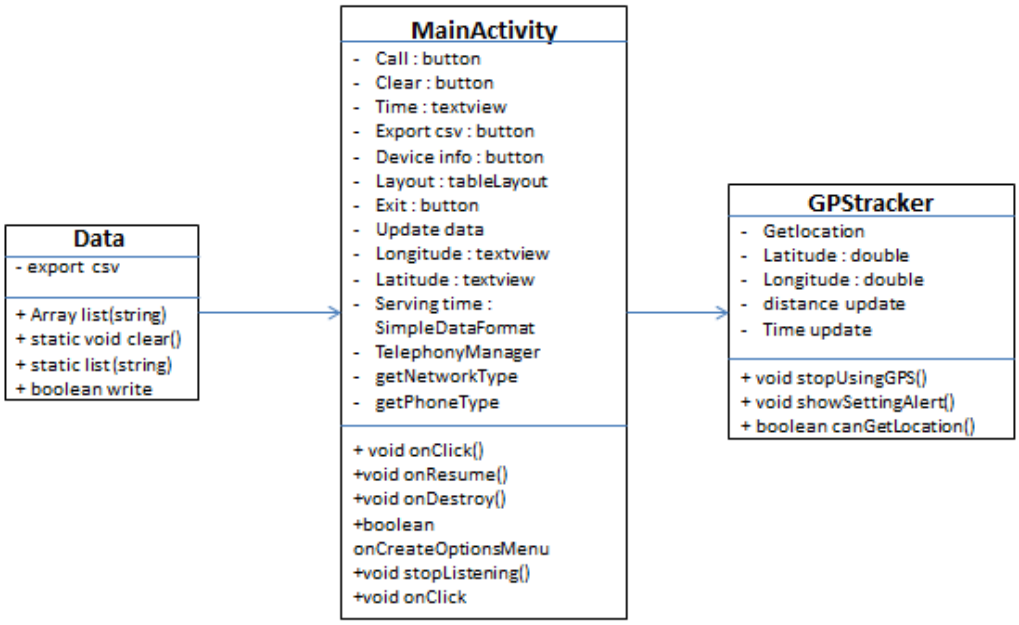
4. *Activity* Diagram *Exit*



Gambar 4.5 *Activity* Diagram *Exit*

4.1.4.3 *Class* Diagram

Class diagram mendeskripsikan struktur dari *class-class* dalam sistem dan mengilustrasikan *operations* serta hubungan antar satu *class* dengan *class* yang lain. Adapun tampilan *class* diagram dari aplikasi Net Info sebgai berikut :



Gambar 4.6 *Class* Diagram Net Info

4.1.4.4 Desain Tabel

Luaran dari aplikasi Net Info berupa CSV dalam bentuk tabel yang nantinya bisa diolah lagi menggunakan *software* pengolah csv. Dengan bentuk tabel maka data yang diperoleh akan bisa diolah menjadi *plotting* map dengan menggunakan *software* bantu MapInfo. Berikut desain dari tabel luaran Net Info :

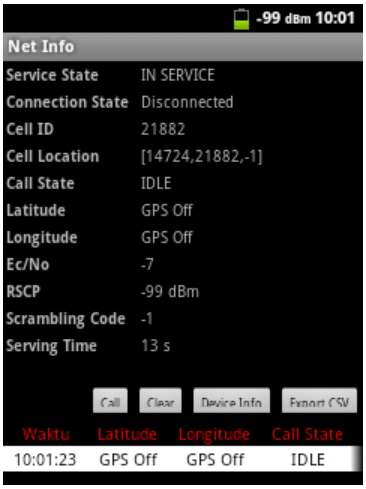
No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	SC	Serving Time
1									
2									

Tabel 4.1 Desain Tabel Luaran Net Info

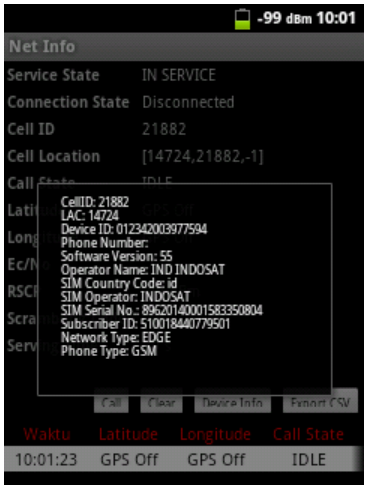
4.1.4.5 Desain *Interface*

Aplikasi Net Info difokuskan untuk *voice call* dimana terdapat menu tombol *call* untuk melakukan panggilan. Luaran dari Net Info berupa .csv yang nantinya bisa di *export* ke MapInfo. Di dalam aplikasi Net Info ini terdapat beberapa parameter untuk *drive test* seperti RSCP, Ec/No, SC, *Serving time, service state*, *longitude* dan *latitude*. Untuk lebih jelasnya, berikut tampilan dari aplikasi Net Info.

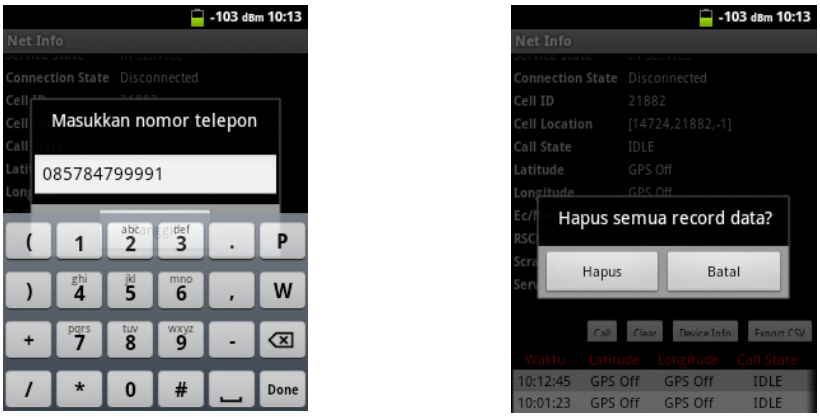
.



(a)



(b)



(c) (d)

Gambar 4.7 Tampilan aplikasi Net Info (a) menu utama

(b) Menu *device info* (c) menu *calling* (d) menu *clear*

4.1.5 *Source Code* Net Info

Aplikasi Net Info ini memiliki 3 *class*, dimana masing-masing *class* memiliki fungsinya sendiri-sendiri. Pada subbab ini akan dijelaskan sebagian *coding* yang ada pada Net Info, untuk *coding* yang lebih lengkap ada pada lampiran A. *Class* pertama yaitu data.java dimana *class* ini berisi *source code* untuk *export* ke *.csv* dan menampilkan *header* dari *serving state*. Berikut adalah *coding* dari cara *export* dan *coding* yang di garis bawah dapat dilihat bahwa data akan di *export* dalam bentuk .csv dengan format hari bulan tahun-jam menit detik.csv.

```
public static boolean write(final String delimiter) {
        boolean status=false;
        File folder = new
File(Environment.getExternalStorageDirectory()+"/NetInfo");
        boolean var = false;
        if (!folder.exists())
            var = folder.mkdir();
```

```
        final String filename = folder.toString()+"/"+new
SimpleDateFormat("ddMMMMyyyy-HHmmss").format(new
Date())+".csv";
        //new Thread() {
            //public void run() {
                try {
                    FileWriter fw =new FileWriter(filename);
                    fw.append("No");
                    fw.append(delimiter);
                    for(int j=1;j<header.length;j++){
                        fw.append(header[j]);
                        fw.append(delimiter);
                    }
                    fw.append('\n');
                    for(int i=0;i<data.nilai.size();i++){
                        fw.append((i+1)+"");
                        fw.append(delimiter);
                        for(int j=1;j<header.length;j++){
                            fw.append(data.nilai.get(i)[j]);
                            fw.append(delimiter);
                        }
                        fw.append('\n');
                    }
                    fw.append("\n");
                    fw.append("Ringing"+delimiter+":
"+ringing+"\n");
                    fw.append("Calling "+delimiter+":
"+calling+"\n");
                    fw.close();
                    status=true;
                } catch (Exception e) {
                }
```

```
            //}
        //}.start();
        return status;
    }
```

	A	B	C	D	E	F	G	H	I	J	K	L
1	No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	Scrambling Code	Serving Time	Jumlah	Event
2	1	-8.79745	115.1612	IDLE	52708817	8042	-5	-81	96	27		null
3	2	-8.79745	115.1612	IDLE	52708817	8042	-8	-81	96	34		null
4	3	-8.79785	115.1609	Calling	52708817	8042	-8	-75	96	27		null
5	4	-8.79785	115.1609	Calling	52708817	8042	-9	-75	96	27		null
6	5	-8.79796	115.1609	Calling	52708817	8042	-9	-75	96	15	69	Succes
7	6	-8.79799	115.1609	Calling	52708817	8042	-7	-75	96	5		null
8	7	-8.79802	115.1609	IDLE	52708817	8042	-7	-79	96	1		null
9	8	-8.79802	115.1609	IDLE	52708817	8042	-7	-80	96	1		null
10	9	-8.79818	115.1608	IDLE	52708817	8042	-10	-77	96	6		null

Gambar 4. Contoh data Net Info dalam bentuk .csv

Gambar di atas metupakan contoh data dari Net Info, dimana setiap kolom berisi nilai sendiri-sendiri. Seperti kolom B dan C berisi *longitude* dan *latitude*, kolom G dan H berisi Ec/No dan RSCP. Selanjutnya data inilah yang akan di olah di MapInfo. *Class* yang ke 2 yaitu GPSTracker.java, untuk *class* ini berisi tentang lokasi *handphone* dengan cara menampilkan fungsi GPS *(Global Potition System)* yang memiliki nilai *longitude* dan *latitude*. Untuk GPS nya menggunakan 2 metode yaitu dengan GPS satelit dan GPS dari jaringan.

```
if (isGPSEnabled) {
  if (location == null) {
    locationManager.requestLocationUpdates(
                LocationManager.GPS_PROVIDER,
                MIN_TIME_BW_UPDATES,
                MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
    if (locationManager != null) {
```

```
location = locationManager.getLastKnownLocation(
        LocationManager.GPS_PROVIDER);
if (location != null) {
        latitude = location.getLatitude();
        longitude = location.getLongitude();
```

Coding diatas merupakan *coding* untuk memfungsikan GPS *handphone* yang berasal dari GPS satelit. Sementara untuk *coding* yang memfungsikan GPS dari jaringan yaitu :

```
if (is NetworkEnabled) {
  if (location == null) {
    locationManager.requestLocationUpdates(
                LocationManager. NETWORK PROVIDER,
                MIN_TIME_BW_UPDATES,
                MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
    if (locationManager != null) {
    location = locationManager.getLastKnownLocation(
          LocationManager.NETWORK _PROVIDER);
    if (location != null) {
          latitude = location.getLatitude();
          longitude = location.getLongitude();
```

Update GPSnya untuk jarak yaitu 1 meter dan waktunya 250ms. Untuk *update*-nya menggunakan *coding* sebagai berikut :

```
Private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES= 1;

private static final long MIN_TIME_BW_UPDATES = 250;
```

Class yang terakhir yaitu Main Activity.java, untuk *class* ini berisi tentang informasi dari *telephony info* seperti RSCP, Ec/No, LAC, Cell id dan seterusnya. Pada *class* ini juga berisi *coding* untuk *device info,* seperti *phone number, software version, operator name, network type* dan seterusnya. Berikut *coding* untuk *telephony info:*

```
@Override
public void onSignalStrengthsChanged(SignalStrength
signalStrength) {
  displayTelephonyInfo();
 kuatlevel = -113+2*(signalStrength.getGsmSignalStrength());
 kualitas =
signalStrength.getGsmSignalStrength()/signalStrength.getEvdoDb
m();
            setTextViewText(R.id.viewRxLev, kuatlevel + "
dBm");
            setTextViewText(R.id.viewRxQual, kualitas + "");
            update_data();
            super.onSignalStrengthsChanged(signalStrength);
```

Dan untuk *coding device info* sebagai berikut :

```
String deviceid = tm.getDeviceId();
String phonenumber = tm.getLine1Number();
String softwareversion = tm.getDeviceSoftwareVersion();
String operatorname = tm.getNetworkOperatorName();
```

```
String simcountrycode = tm.getSimCountryIso();
String simoperator = tm.getSimOperatorName();
String simserialno = tm.getSimSerialNumber();
String subscriberid = tm.getSubscriberId();
String networktype = getNetworkTypeString(tm.

getNetworkType());
```

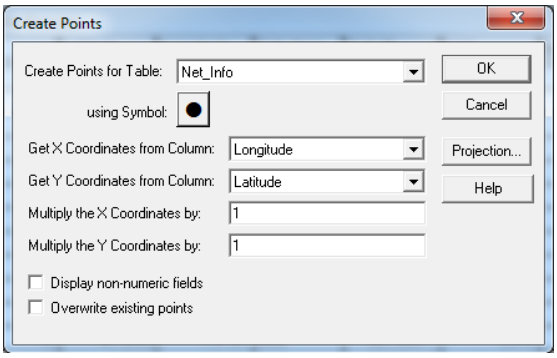
4.2 Pengolahan Data

Data yang didapatkan dari hasil *export* menggunakan Net Info selanjutnya akan diolah menggunakan *software* pengolah CSV seperti MS Office Excel, dan untuk melihat sebaran dari RSCP, Ec/No dan SC bisa menggunaka *software* MapInfo Professional 11.

4.2.1 *Plotting* Data Net Info ke MapInfo

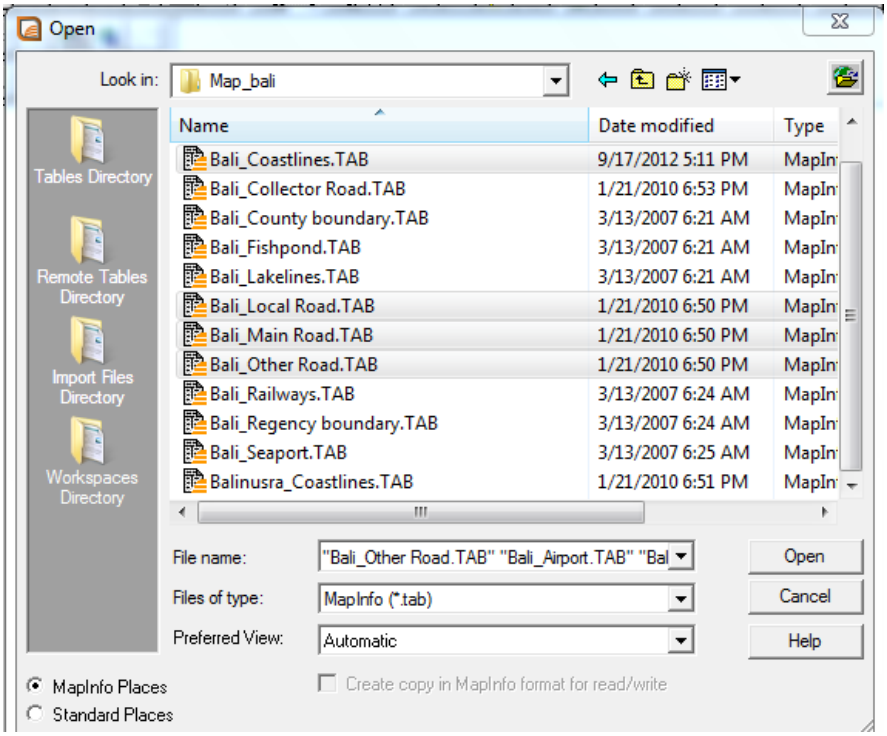
Data luaran dari aplikasi Net Info adalah berupa .CSV yang nantinya bisa di di *software* MS Office Excel, MapInfo, maupun *software* pengolah .CSV lainya. Untuk pada Tugas Akhir ini, data akan diolah di *software* MapInfo untuk menampilkan data berupa *plotting* di map/peta. Berikut langkah-langkah *plotting* data Net Info untuk mengetahui nilai RSCP :

- Buka MapInfo
- Untuk membuka data Net Info, pilih menu *file – open*. Untuk *file of type* pilih *comma delimited CSV (*.CSV)*
- Setelah tampil tabel dari data Net Info, pilih menu *Table – Create Points.*
- Lalu isi seperti gambar di bawah ini.



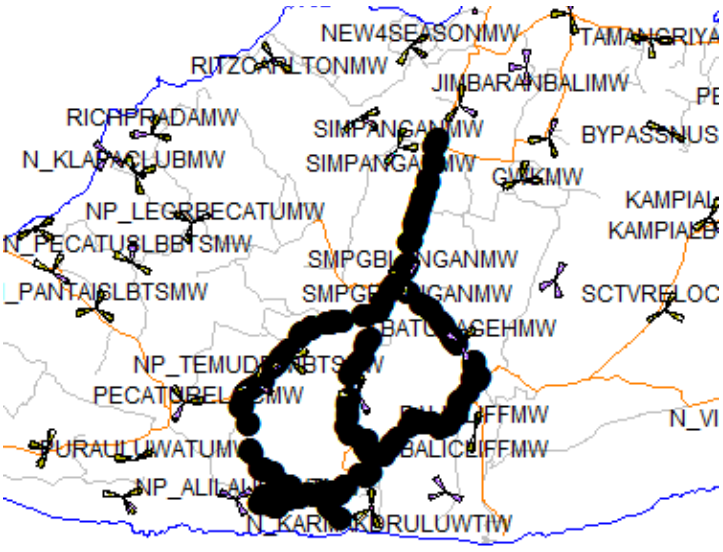
Gambar 4.9 *Create Points*

- Untuk menampilkan *map* dari tabel, pilih menu *Windows – New Map Windows.*
- Tambahkan *map* Bali dan *site B* dengan cara pilih *file –open.* Pilih *map* yang akan di buka.



Gambar 4.10 *Open Map* Bali

- Hasilnya akan seperti ini.

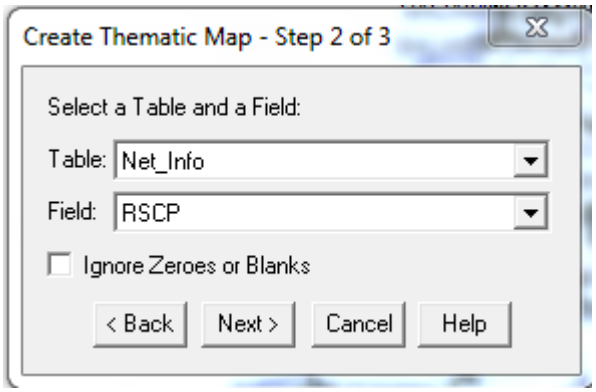


Gambar 4.11 *Plotting* hasil dari CSV

4.2.2 *Plotting* Data Net Info Berdasarkan Nilai RSCP

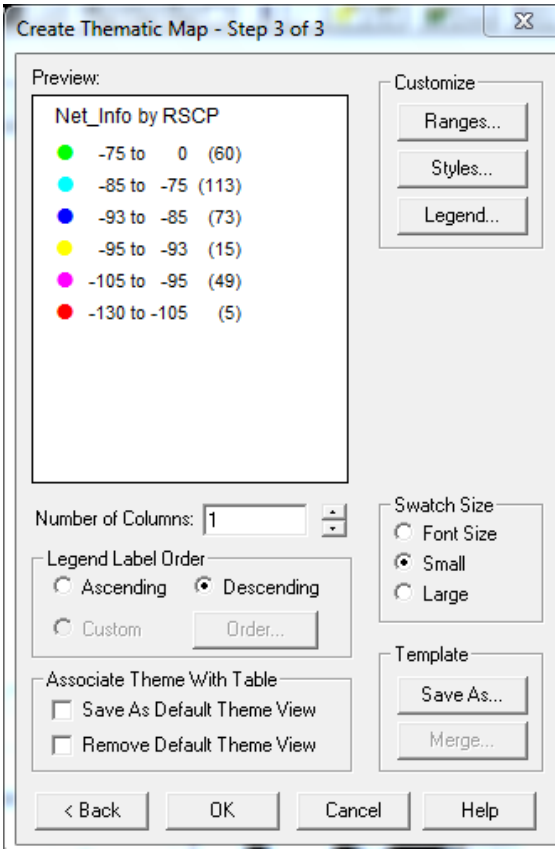
Setelah data bisa ditampilkan di *map*, maka langkah selanjutnya adalah menampilkan data berdasarkan nilai RSCP. Langkah-langkahnya sebagai berikut :

- Pilih menu *Map – Create Thematic Map.*
- Pilih tipe dari *template,* lalu *Next.*
- Isi *Table* : Net_Info dan *Field :*RSCP, lalu *Next.*



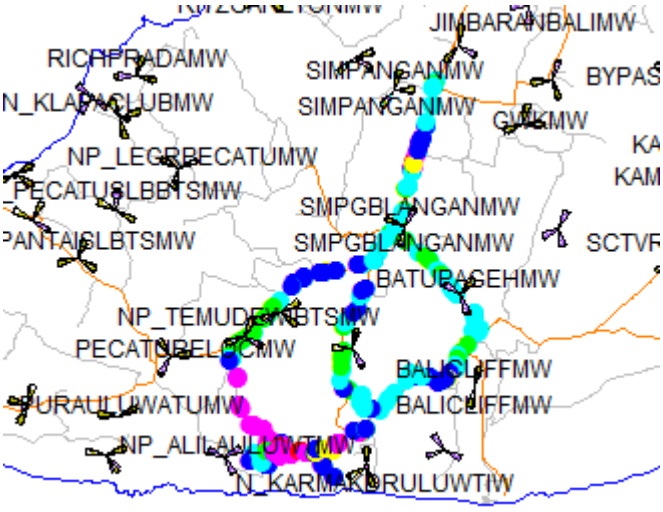
Gambar 4.12 *create thematic map* RSCP

- Isi *range* nilai beserta tipe warna sesuai gambar di bawah :



Gambar 4.13 *Range* dari *thematic map* RSCP

- Lalu OK, dan hasilnya akan seperti ini.

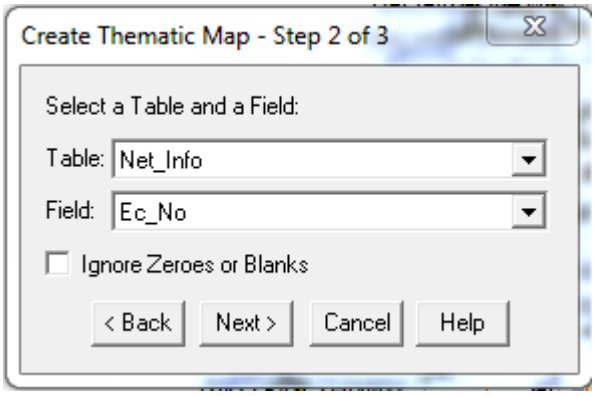


Gambar 4.14 Hasil *export* data RSCP

4.2.3 *Plotting* Data Net Info Berdasarkan Nilai EcNo

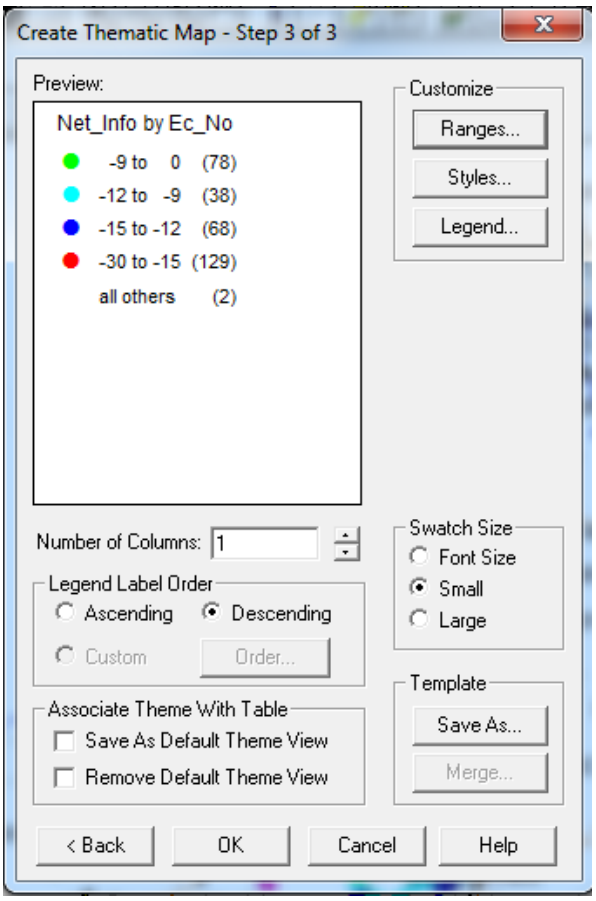
Setelah *plotting* nilai RSCP, selanjutnya *plotting* data Net Info berdasarkan nilai EcNo. Caranya hampir sama hanya berbeda pada masukan *Field* dan *range* nilainya. Berikut langkah-langkahnya :

- Pilih menu *Map – Create Thematic Map.*
- Pilih tipe dari *template,* lalu *Next.*
- Isi *Table* : Net_Info dan *Field* :Ec_No, lalu *Next.*



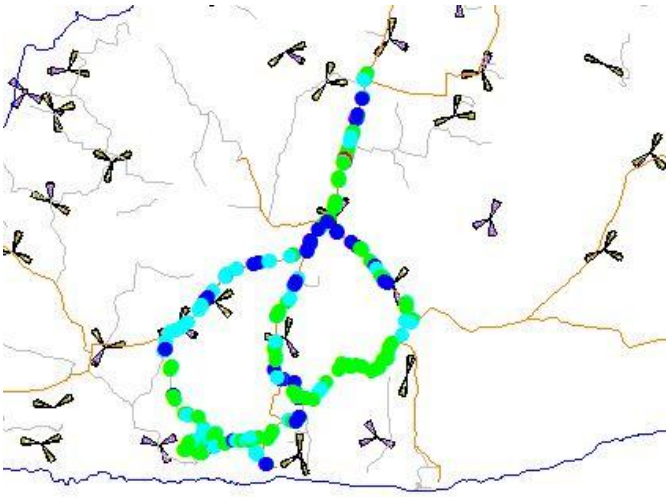
Gambar 4.15 *create thematic map* Ec/No

- Isi *range* nilai beserta tipe warna sesuai gambar di bawah :



Gambar 4.16 *Range* dari *thematic map* Ec/No

- Lalu OK, dan hasilnya akan seperti ini.



Gambar 4.17 Hasil *export* data Ec/No

4.3 Analisis Data Hasil *Drive Test*

Berdasarkan hasil *drive test* menggunakan Net Info didapatkan sebanyak 315 data yang berisi tentang info *longitude*, *latitude*, nilai RSCP, nilai Ec/No, *Scrambling Code, Cell Id,* serta *serving time.* Dari data tersebut tergolong sedikit namun masih bisa mengcover semua area. Hal ini dikarenakan saat pengambilan data menggunakan Net Info menggunakan metode *Calling* atau menelfon, sehingga GPS *Handphone* tidak bekerja secara maksimal. Namun dengan metode ini dapat sekaligus melihat terjadinya *drop call* dan *block call.* Untuk selanjutnya kedua data akan dibandingkan lebih detail berdasarkan nilai RSCP, Ec/No dan *Scrambling Code.*

4.3.1 Analisis Nilai RSCP

Received Signal Code Power dapat digunakan untuk menganalisis *coverage* dari *site* yang di *drive test.* Tidak ada standar yang ditetapkan untuk nilai RSCP. Setiap operator memiliki ambang yang berbeda-beda. Nilai RSCP yang digunakan pada Tugas Akhir ini berdasarkan *drive test report* PT Cahya Mitratama *Technology*

dan dapat dilihat pada tabel 4.1. Tabel 4.2 merupakan hasil *drive test* sebagian data Net Info pada tanggal 20 Agustus 2013 pukul 16:26:00 WITA. Dari tabel itu dapat dilihat nilai RSCP di tempat dengan keterangan *longitude* dan *latitude*nya. Setiap tempat memiliki nilai RSCP berbeda-beda sesuai dengan banyak faktor mulai dari kontur tanah sampai konfigurasi *tilting* antena. Dari tabel 4.2 nilai RSCP paling baik yaitu -83dbm pada *longitude* 115.1358 dan *latitude* -8.83155, sedangkan nilai RSCP paling buruk yaitu -105dbm. Berdasarkan *range* nilai RSCP pada tabel 4.1, maka nilai -83dbm termasuk dalam *range* warna hijau muda dan nilai -105dbm termasuk dalam warna merah. Untuk melihat semua data RSCP pada Tugas Akhir ini ada pada lampiran B dan data yang berupa tabel tersebut akan lebih jelas dalam menganalisanya maka harus di *plotting* ke MapInfo. Pada gambar 4.12 merupakan hasil *plotting* ke MapInfo berdasarkan nilai RSCP.

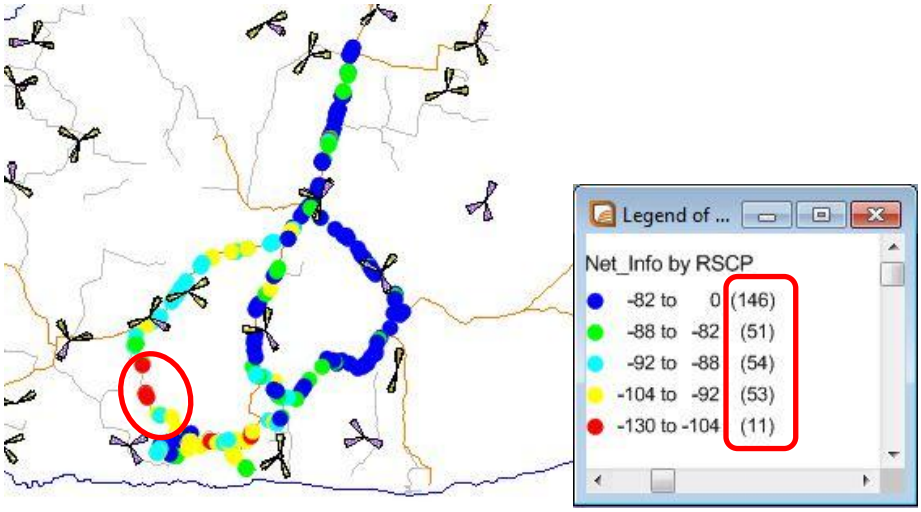
Standar warna	Range nilai	Keterangan
	-82 sampai 0	Sangat baik
	-88 sampai -82	Baik
	-92 sampai -88	Sedang
	-104 sampai -92	Buruk
	-130 sampai -104	Sangat buruk

Tabel 4.2 *Range* nilai RSCP

No	Latitude	Longitude	**Call State**	**Cell ID**	**LAC**	**Ec/No**	RSCP	**Serving Time**
1	**-8.83155**	**115.1358**	IDLE	52571985	8011	-15	**-83**	6
2	**-8.83171**	**115.1357**	IDLE	52571985	8011	-15	**-83**	11
3	**-8.83174**	**115.1357**	Calling	52571985	8011	-12	**-89**	1
4	**-8.83362**	**115.1366**	Calling	52571985	8011	-12	**-89**	67

5	**-8.83373**	**115.1365**	IDLE	52571985	8011	-4	**-105**	2
6	**-8.83388**	**115.1365**	IDLE	52571985	8011	-8	**-97**	3
7	**-8.83404**	**115.1365**	IDLE	52574944	8011	-4	**-105**	3
8	**-8.83725**	**115.1368**	Calling	52574944	8011	-4	**-105**	70
9	**-8.8373**	**115.1368**	IDLE	52574944	8011	-7	**-99**	1
10	**-8.83744**	**115.1369**	IDLE	52574944	8011	-4	**-105**	2
11	**-8.83774**	**115.1371**	IDLE	52574944	8011	-4	**-105**	8
12	**-8.83959**	**115.1382**	Calling	52574944	8011	-7	**-99**	60
13	**-8.83968**	**115.1385**	Calling	52574944	8011	-7	**-99**	8
14	**-8.83973**	**115.1386**	IDLE	52574944	8011	-7	**-99**	9

Tabel 4.3 Data *export* 20August2013-162600



Gambar 4.18 *Spot* nilai RSCP buruk

Dari hasil *plotting* data berdasarkan nilai RSCP, dapat dilihat bahwa terdapat spot-spot dengan nilai RSCP yang sangat bagus sampai yang sangat buruk. Dari gambar tersebut juga dapat dilihat jumlah total *spot* untuk masing-masing kategori, sehingga dapat diketahui prosentasenya dengan rumus 4.1. Prosentase untuk nilai RSCP dengan ketegori sangat baik yaitu :

$$Rasio\ Spot\ sangat\ bagus = \frac{\Sigma \text{spot sangat bagus}}{\Sigma \text{spot total}} x\ 100\% \dots\dots\dots\dots (4.1)$$

$$\frac{146}{315} x100\% = 46{,}34\ \% \dots\dots\dots\dots\dots (4.2)$$

Menggunakan perhitungan yang sama maka untuk prosentase RSCP dengan kategori baik yaitu 16,19%. Untuk sedang sebesar 17,14%, kategori buruk 16,82% dan kategori sangat buruk hanya 3,49%. Sehingga untuk area yang memiliki *low coverage* terdapat pada kategori buruk dan sangat buruk, dengan prosentase mencapai 20,31%. Ketentuan *threshold* untuk RSCP baik dari *provider* yaitu 95%. Untuk DTR (*Drive Test Report*) dan perhitungan KPI terdapat di lampiran c. Berdasarkan perhitungan data diatas, nilai RSCP baik hanya mencapai 79,68% sehingga pada area tersebut RSCP belum mancapai target dari *provider*. Pada gambar 4.12 spot-spot yang memiliki nilai RSCP buruk ada ditandai dengan lingkaran merah. Pada spot tersebut memiliki nilai RSCP buruk karena pada area/spot tersebut berupa dataran tinggi dan berupa hutan belantara. Meskipun memiliki RSCP yang buruk namun tidak menjadi banyak masalah karena pada area tersebut tidak banyak pengguna telepon. Berikut foto dari *spot* yang memiliki RSCP buruk.

Gambar 4.19 Lokasi nilai RSCP buruk

4.3.2 Analisis Nilai Ec/No

Ec/No adalah rasio perbandingan antara energi yang dihasilkan dari sinyal pilot dengan total energi yang diterima. Ec/No juga menunjukkan level daya minimum (*threshold*) dimana MS masih bisa melakukan suatu panggilan. Atau dengan kata lain, Ec/No adalah kualitas data atau suara di jaringan operator 3G/UMTS. Fungsinya sama dengan RxQual di jaringan 2G. Ec/No adalah perbandingan antara energi setiap *chip* sinyal informasi terhadap sinyal interferensi atau sinyal derau (*noise*) yang menyertainya. Pada intinya adalah perbandingan antara kuat sinyal yang dikehendaki terhadap kuat sinyal yang tidak dikehendaki. Makin besar nilai Ec/No akan memberikan performansi yang lebih baik.

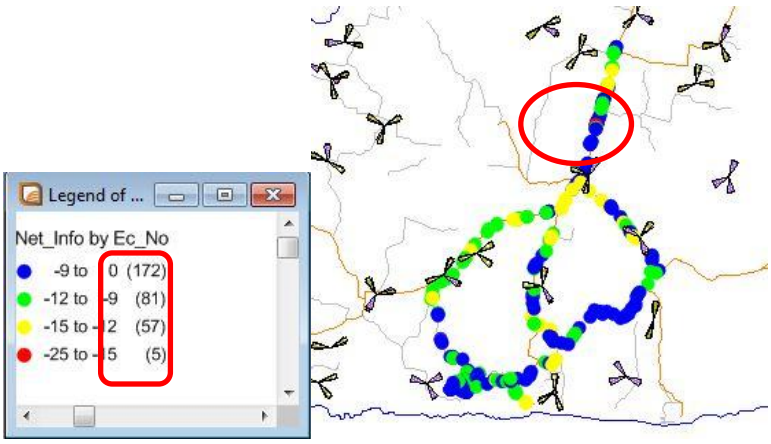
Tidak ada standar yang ditetapkan untuk nilai Ec/No. Setiap operator memiliki ambang yang berbeda-beda. Nilai Ec/No yang digunakan pada Tugas Akhir ini dapat dilihat pada Tabel 4.4. Tabel 4.5 merupakan hasil *drive test* sebagian data Net Info pada tanggal 20 Agustus 2013 pukul 15:33:16 WITA. Seperti pada analisis RSCP, pada tabel 4.5 dapat dilihat nilai Ec/No di tempat dengan keterangan *longitude* dan *latitude*nya. Dari tabel tersebut nilai Ec/No paling buruk yaitu -25dbm. Berdasarkan *range* nilai Ec/no pada tabel 4.3, maka nilai -25dbm termasuk dalam *range* warna merah dan merupakan kategori sangat buruk sehingga. Data keseluruhan dari Ec/No ada pada lampiran B dan untuk lebih jelas maka data yang berupa tabel tersebut akan di *plotting* ke MapInfo. Data tabel Ec/No secara keseluruhan dapat dilihat pada lampiran B. Pada gambar 4.14 merupakan hasil *plotting* ke MapInfo berdasarkan nilai Ec/No pada spot kategori yang sangat buruk.

	-9 sampai 0	Sangat baik
	-12 sampai -9	Baik
	-15 sampai -12	Sedang
	-25 sampai -15	Buruk

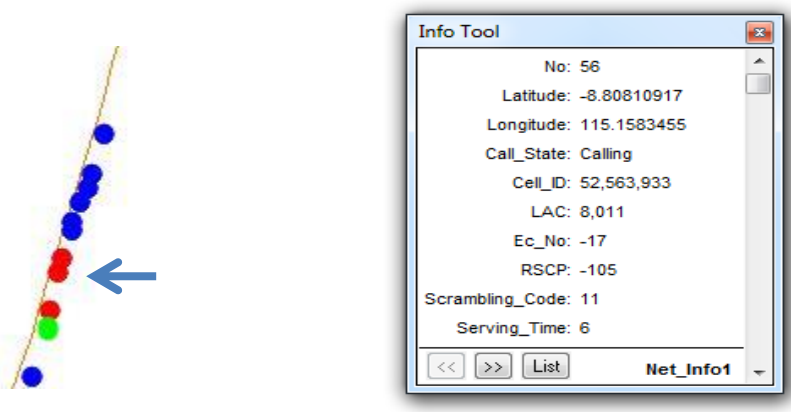
Tabel 4.4 *Range* nilai Ec/No

No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	Serving Time
50	**-8.80785**	**115.1584**	IDLE	52563933	8042	**-6**	-85	5
51	**-8.80789**	**115.1584**	IDLE	52563933	8042	**-7**	-85	14
52	**-8.80789**	**115.1584**	IDLE	52563933	8011	**-8**	-85	14
53	**-8.80804**	**115.1584**	IDLE	52563933	8011	**-8**	-80	35
54	**-8.80804**	**115.1584**	IDLE	52563933	8011	**-16**	-80	35
55	**-8.80811**	**115.1583**	IDLE	52563933	8011	**-16**	-97	2
56	**-8.80811**	**115.1583**	Calling	52563933	8011	**-17**	-105	6
57	**-8.80832**	**115.1583**	IDLE	52563933	8011	**-25**	-105	9
58	**-8.80832**	**115.1583**	IDLE	52563933	8011	**-22**	-91	9
59	**-8.80841**	**115.1583**	IDLE	52563933	8011	**-11**	-91	5
60	**-8.80843**	**115.1583**	IDLE	52563933	8011	**-11**	-91	5
61	**-8.80868**	**115.1582**	Calling	52563933	8011	**-5**	-85	10
62	**-8.80868**	**115.1582**	Calling	52563933	8011	**-5**	-85	10
63	**-8.81038**	**115.1576**	Calling	52563933	8011	**-6**	-85	51

Tabel 4.5 Data *export* 20August2013-153316



(a)



(b)

Gambar 4.20 *Spot* nilai Ec/No buruk (a) *Spot* Ec/No dan *thematic* (b) *spot* Ec/No diperbesar dan *Info tool*

Dari hasil *plotting* data berdasarkan nilai Ec/No, dapat dilihat bahwa terdapat spot dengan nilai Ec/No yang sangat bagus sampai yang sangat buruk. Prosentase untuk nilai RSCP dengan ketegori sangat baik yaitu :

$$Rasio\ Spot\ sangat\ baik = \frac{\Sigma \text{spot sangat baik}}{\Sigma \text{spot total}} x\ 100\% \dots\dots\dots\dots (4.3)$$

$$\frac{172}{315} x 100\% = 54{,}6\ \% \dots\dots\dots\dots (4.4)$$

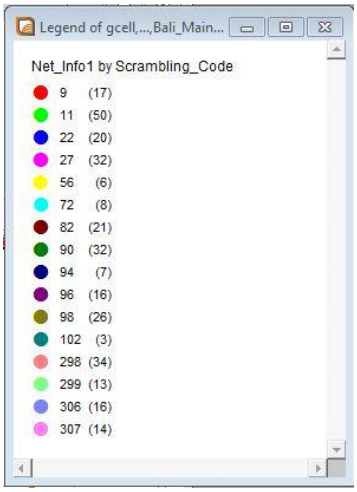
Menggunakan perhitungan yang sama maka untuk prosentase Ec/No dengan kategori baik yaitu 25,71%. Untuk kategori sedang 18,09% dan kategori buruk hanya 1,58%. Ketentuan *threshold* untuk Ec/No baik dari *provider* yaitu 90%. Untuk DTR (*Drive Test Report*) dan perhitungan KPI terdapat di lampiran c. Berdasarkan perhitungan data diatas, nilai Ec/No baik mencapai 98,41% sehingga pada area tersebut Ec/No sudah mencapai target dari *provider*. Pada gambar 4.14 spot-spot yang memiliki nilai Ec/No buruk ada ditandai dengan lingkaran merah. Pada spot tersebut yang letaknya pada *latitude -8*.80832116 dan *longitude 115*.1582955 memiliki nilai Ec/No -25, hal

ini dapat dikatakan bahwa nilai Ec/No buruk pada spot tersebut. Dari hasil pengamatan di lapangan, spot tersebut merupakan jalan menanjak yang cukup curam sehingga sinyal tidak terlalu bagus. Namun secara keseluruhan, pada area *drive test* Uluwatu ini dapat dikatakan kualitas sinyalnya bagus untuk melayani panggilan suara. Berikut foto pada spot tersebut.

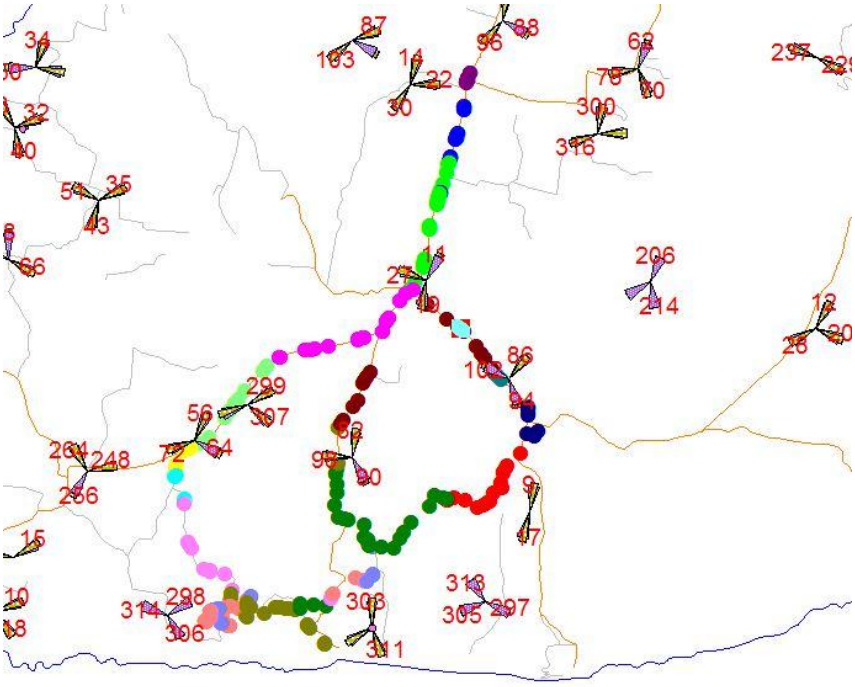
Gambar 4.21 Lokasi nilai Ec/No buruk

4.3.3 Analisis *Scrambling Code* (SC)

Scrambling Code berguna untuk membedakan UE yang satu dengan UE yang lain disisi uplink dan juga untuk membedakan *node B* yang satu dengan *node B* yang lainnya disisi downlink. Sehingga SC memberikan informasi sektor dari site yang sedang *serving*. Berikut *plotting* data berdasarkan nilai *scrambling code.*



Gambar 4.22 *Thematic Scrambling code*



Gambar 4.23 Hasil *plotting* SC

Gambar 4.16 menunjukkan tipe warna dari kode nomer setiap *sector* antena. Jadi untuk cara membacanya hanya dengan melihat warna yang ada pada *plotting* data pada gambar 4.17 dan disamakan dengan warna yang ada di tabel *legend* gambar 4.16. Dari *plotting* sebaran SC terlihat masing-masing sektor telah mengcover area dengan SC yang telah ditentukan. Ini menunjukkan tidak terjadi *cross feeder*. *Cross feeder* adalah suatu kondisi dimana terjadi kesalahan pemasangan kabel *feeder* pada antena. Hal ini dapat mengakibatkan kesalahan pembacaan monitoring dari BSC.

4.3.4 Analisis CSSR dan DCR

Analisis CSSR (*Call Setup Success Rate*) dan DCR (*Drop Call Rate*) dilakukan untuk mengetahui seberapa besar kualitas pelayanan yang disediakan

oleh operator yang bersangkutan. Dari data tabel dapat diperoleh keterangan untuk mengetahui jumlah *drop calls*, *block calls* dan *success calls.*

Pada Tugas Akhir ini, waktu dalam melakukan *calling* yaitu minimal 1 meni, dan dilakukan secara terus menerus sampai proses *drive test* selesai. Dari hasil perhitungan diperoleh nilai sukses panggilan sebanyak 49, *block call* sebanyak 1 dan *drop call* memiliki nilai 0. Sehingga untuk prosentasenya sebagai berikut :

$$CSSR = \frac{Succes\ Call}{Call\ Attems} x100\% = \frac{49}{50} x100\% = 98\% \dots\dots\dots (4.3)$$

$$DCR = \frac{Call\ Drop}{Call\ Attems} x\ 100\% = \frac{0}{50} = 0\% \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (4.4)$$

$$BCR = \frac{Call\ Block}{Call\ Attems} x\ 100\% = \frac{1}{50} = 2\% \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (4.5)$$

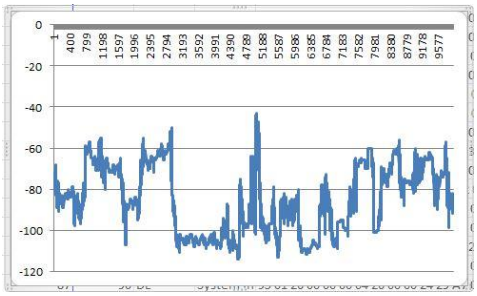
Dari perhitungan nilai CSSR (*Call Setup Success Rate*), DCR (*Drop Call Rate*) dan BCR (*Block Call Rate*) tersebut dapat diketahui bahwa kualitas pelayanan yang disediakan oleh operator sangat bagus yang ditunjukkan oleh nilai CSSR sebesar 98% dan DCR sebesar 0%, yang berarti panggilan berjalan sangat lancar tanpa adanya gangguan dropped calls. Ketentuan *drop call* dari *provider* yaitu 0%, sehingga pada area ini sudah memenuhi permintaan dari *provider* tersebut. Ketentuan ini dapat dilihat pada lampiran c. Namun masih terjadi 1 kali *block call* yang mengganggu performansi jaringan. Dari hasil data dapat dilihat bahwa *block call* terjadi pada *longitude* 115.1583 *latitude* -8.80811 dan jika dilihat pada nilai Ec/No sebesar -17 dan RSCP -105. Dari nili Ec/No dan RSCP tersebut dapat disimpulkan bahwa pada titik tersebut memiliki sinyal yang sangat buruk. Hal ini karena pada titik tersebut merupakan jalan menanjak yang cukup curam seperti dijelaskan pada analisis Ec/No.

4.4 Membandingkan Data Net Info dengan TEMS

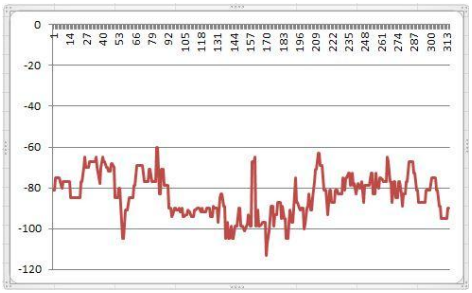
Data yang diperoleh ketika *drive test* menggunakan Net Info berjumlah 315, sementara jika menggunakan TEMS Investigation data yang diperoleh sebanyak 9983 data. Dapat dikatakan ketika menggunakan Net Info data merupakan sampling 31,69 x dari data TEMS Invesigation.

$$\frac{Data\ TEMS}{Data\ Net\ Info} = \frac{9983}{315} = 31{,}69 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4.6)$$

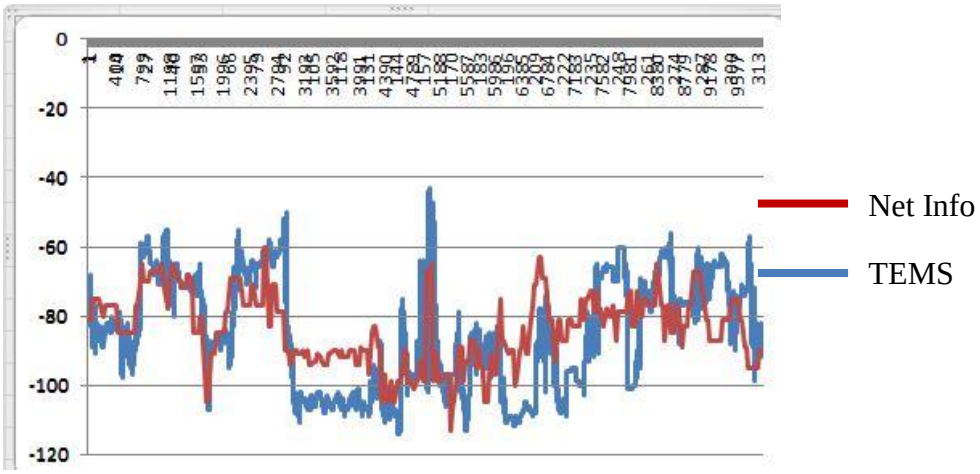
Berikut adalah grafik data RSCP yang diperoleh menggunakan TEMS, grafik data RSCP menggunakan Net Info dan ketika kedua grafik dari TEMS dan Net Info di gabungkan, hasilnya seperti berikut:



(a)



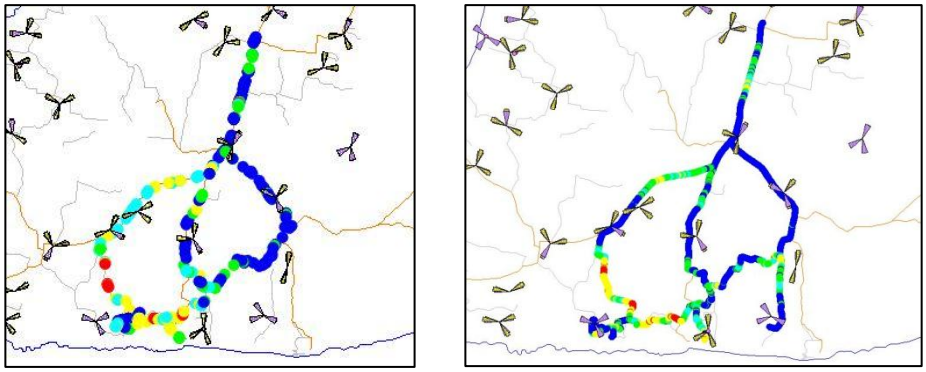
(b)



(c)

Gambar 4.24 Grafik RSCP (a) Grafik RSCP TEMS (b) Grafik RSCP Net Info

(c) Grafik RSCP gabungan

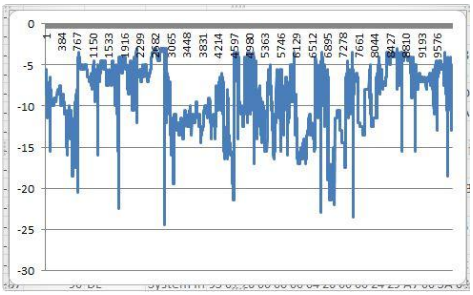


(a) (b)

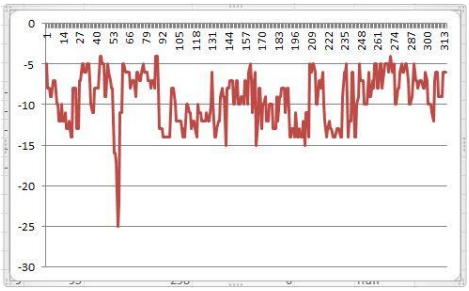
Gambar 4.25 Perbandingan data RSCP berdasarkan *plotting* di *map*

(a) *Plotting* RSCP Net Info (b)*Plotting* RSCP TEMS

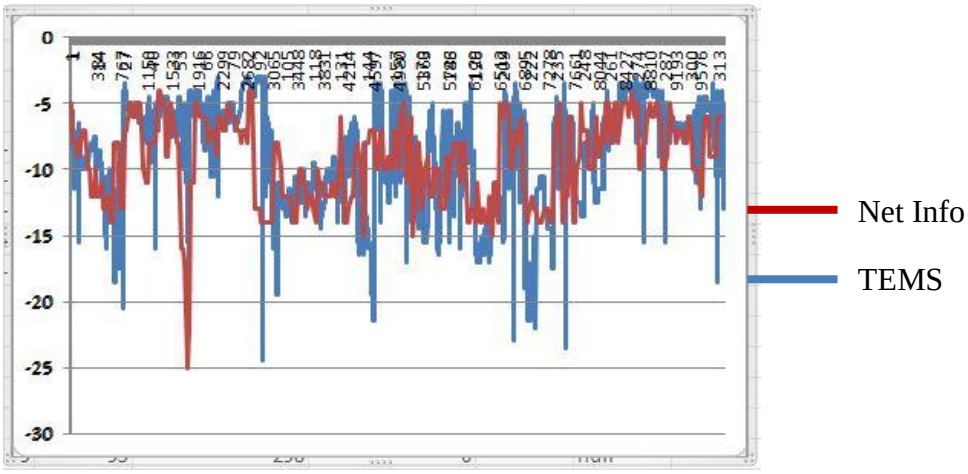
Dari grafik di atas, dapat dikatakan bahwa grafik dari Net Info hampir sama dengan grafik TEMS. Terdapat perbedaan sedikit yang bisa disebabkan faktor kekuatan handphone dalam menerima sinyal. Selanjutnya jika data dibedakan dengan cara *plotting* gambar berdasarkan RSCP hasilnya sama dengan saat data dibedakan dengan grafik, perbedaan data tidak terlalu jauh. Perbedaan yang ada hanya pada kerapatan titiknya, dimana data Net Info tidak terlalu rapat titik-titik warnanya. Untuk perbandingan selanjutnya yaitu membandingkan data berdasarkan Ec/No. Pada gambar 4.20c merupakan grafik jika Ec/No keduanya digabungkan dan hasilnya juga tidak terlalu berbeda, begitu pula pada *plotting* Ec/No pada gambar 4.21. Sehingga dapat dikatakan bahwa data Net Info sama dengan data dari TEMS.



(b)



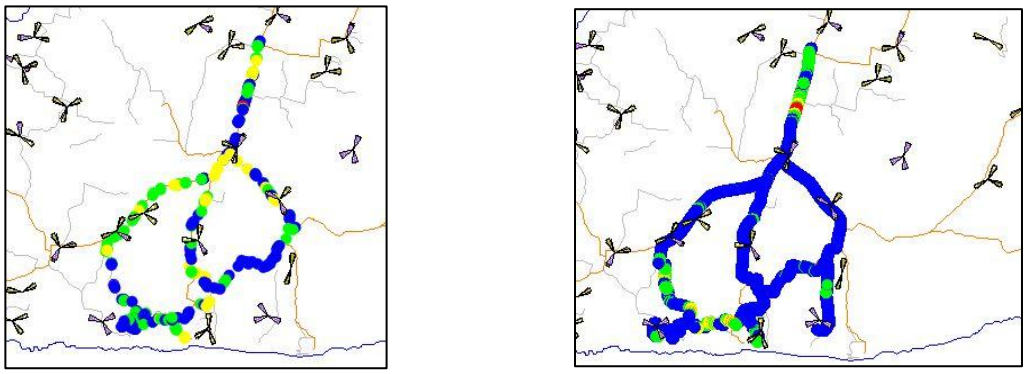
(b)



(c)

Gambar 4.26 Grafik Ec/No (a) Grafik Ec/No TEMS (b) Grafik Ec/No Net Info

(c) Grafik Ec/No gabungan



(a) (b)

Gambar 4.27 Perbandingan data Ec/No berdasarkan *plotting* di *map*

(a) *Plotting* Ec/No Net Info (b)*Plotting* Ec/No TEMS

BAB 5. PENUTUP

1.1 Kesimpulan

Berdasarkan proses yang telah dilakukan pada Tugas Akhir ini, dapat disimpulkan beberapa hal, antara lain :

1. a. Kekuatan sinyal (RSCP) dari *cluster* Uluwatu masih buruk, dimana untuk nilai RSCP baik hanya 79,68% dan belum mencapai target *threshold* dari *provider* yaitu 95% (Halaman 47).

   b. Nilai Ec/No dari *cluster* Uluwatu cukup baik, dimana untuk nilai Ec/No baik mencapai 98,41% dan telah mencapai target *threshold* dari *provider* yaitu 90% (Halaman 50).

   c. Kualitas layanan *voice call* jika dilihat dari nilai CSSR dan DCR sudah sangat baik. Untuk CSSR mencapai 98%, hanya terjadi 1 kali *block call* untuk 50 panggilan yang dilakukan. Sementara itu untuk *drop call* memiliki prosentase 0% yang artinya tidak terjadi *drop call* selama *drive test* dan sudah mencapai target dari *provider* yaitu 0% (Halaman 53).

2. *Drive test* dengan *software* Net Info bisa mengcakup semua area namun tingkat ketelitian sangat kecil jika dibandingkan dengan *drive test* menggunakan *software* TEMS Investigations yaitu 1 : 32 (Halaman 54).

3. Perbandingan *drive test* menggunakan Net Info dengan menggunakan TEMS Investigation jika dilihat dari hasil grafik maupun *plotting* data berdasarkan RSCP, Ec/No dan *Scrambling code* maka hasilnya mirip (Halaman 55).

1.1.1 Saran

Aplikasi ini tentu saja masih belum sempurna, banyak yang dapat dilakukan untuk mengembangkan aplikasi ini agar menjadi lebih baik lagi, antara lain :

1. Perlu pengembangan untuk aplikasi Net Info sehingga tidak hanya untuk *drive test voice call* tapi bisa untuk paket data internet, sehingga bisa digunakan untuk menganalisa kecepatan *download* dari suatu *site-B* atau *cluster*.
2. Pada aplikasi ini perlu pengembangan dalam masalah *drive test single site* karena belum ada fitur *lock site* atau BTS.

DAFTAR PUSTAKA

Admin, 2008. *Konsep Dasar Jaringan WCDMA-UMTS.* http://digilib.ittelkom.ac.id/index.php?option=com_content&view=article&id=349:konsep-dasar-jaringan-wcdma-umts&catid=17:sistem-komunikasi-bergerak&Itemid=14 [22 Mei 2013]

Arifin, Muhammad Zaenal. 2012. *Sistem Informasi Geografis Untuk Fasilitas Perguruan Tinggi Berbasis Android di Kota Surabaya.* Politeknik Elektronika Negeri Surabaya ITS. Surabaya.

Huda, Arif Akbarul. 2012. *24 Jam!! Pintar Pemrograman Android*. C.V Andi Offset. Yogyakarta.

Kusuma, R Bram Aditya. 2011. *Analisis Kualitas Voice Call Pada Jaringan WCDMA Menggunakan Tems Investigation.* Universitas Diponegoro. Semarang.

Naufal, Herdi. 2012. *Mengenal Arsitektur Android OS.* http://www.twoh.web.id/2012/09/mengenal-arsitektur-sistem-operasi-android/ [10 April 2013]

Novrizal, Zaimi. 2011. *Analisa Performansi Jaringan 3G Untuk Optimasi Jaringan.* Universitas Sumatera Utara. Medan.

PT. Cahya Mitratama Technology. 2013. *3G Drive Test VIP Complaint Area Tol Nusadua.* Denpasar. Nokia Siemens Networks NPO Sub Region Indonesia.

Siboro, Pebriantono. 2011. *Analisis Performansi Sinyal GSM Dengan Optimasi Tilting Antena BTS Berdasarkan Drive Test.* Universitas Sumatera Utara. Medan.

Supardi, Yuniar. 2011. *Semua Bisa Menjadi Programmer Android Basic*. PT Elex Media Komputindo. Jakarta.

Wardhana, Lingga. 2011.*2G/3G RF Planning and Optimization for Consultant.*www.nulisbuku.com.Jakarta Selatan

Wibisono, Gunawan. Usman, Uke Kurniawan. Hantoro, Gunadi Dwi. 2007. *Konsep Teknologi Seluler.* Informatika. Bandung.

Lampiran A. *Coding* dari aplikasi Net Info

1. *Source Code class* data.java

```
package com.DT.netinfota;

import android.os.Environment;

import java.io.File;

import java.io.FileWriter;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;


/**

 * Created by Agung on 06/08/13.

 */

public class data {

    public static List<String[]> nilai = new
ArrayList<String[]>();

    public static int ringing=0,calling=0,idle=0;

    public static String header[]={"Waktu",

            "Latitude",

            "Longitude",

            "Call State",

            "Cell ID",

            "LAC",

            "Ec/No",
```

```
            "RSCP",
            "Scrambling Code",
            "Serving Time",
    };
    public static void clear(){
        idle=0;
        ringing=0;
        calling=0;
        nilai.clear();
    }
    public static boolean write(final String delimiter) {
        boolean status=false;
        File folder = new
File(Environment.getExternalStorageDirectory()+"/NetInfo");
        boolean var = false;
        if (!folder.exists())
            var = folder.mkdir();
        final String filename = folder.toString()+"/"+new
SimpleDateFormat("ddMMMMyyyy-HHmmss").format(new
Date())+".csv";
        //new Thread() {
        //public void run() {
        try {
            FileWriter fw = new FileWriter(filename);
            fw.append("No");
            fw.append(delimiter);
```

```
            for(int j=1;j<header.length;j++){
                fw.append(header[j]);
                fw.append(delimiter);
            }
            fw.append('\n');
            for(int i=0;i<data.nilai.size();i++){
                fw.append((i+1)+"");
                fw.append(delimiter);
                for(int j=1;j<header.length;j++){
                    fw.append(data.nilai.get(i)[j]);
                    fw.append(delimiter);
                }
                fw.append('\n');
            }
            fw.append("\n");
            fw.append("Ringing "+delimiter+":
"+ringing+"\n");
            fw.append("Calling "+delimiter+":
"+calling+"\n");
            fw.close();
            status=true;
        } catch (Exception e) {
        }
        //}
        //}.start();
```

```
        return status;

    }

}
```

2. *Source Code class* GPSTracker.java

```
package com.DT.netinfota;
/**
 * Created by Agung on 8/06/13.
 */
import android.app.AlertDialog;
import android.app.Service;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.os.IBinder;
import android.provider.Settings;

public class GPSTracker extends Service implements
LocationListener {

    private final Context mContext;
    private MainActivity view;
    boolean isGPSEnabled = false;
    boolean isNetworkEnabled = false;
    boolean canGetLocation = false;
    Location location;
    public static double latitude;
    public static double longitude;

    private static final long
MIN_DISTANCE_CHANGE_FOR_UPDATES = 1;
    private static final long MIN_TIME_BW_UPDATES = 250;
    protected LocationManager locationManager;
    public GPSTracker(Context context,MainActivity view)
{
```

```
        this.mContext = context;
        this.view = view;
        getLocation();
    }
    public Location getLocation() {
        try {
            locationManager = (LocationManager)
mContext.getSystemService(LOCATION_SERVICE);
            isGPSEnabled =
locationManager.isProviderEnabled(LocationManager.GPS_PRO
VIDER);
            isNetworkEnabled =
locationManager.isProviderEnabled(LocationManager.NETWORK
_PROVIDER);
            if (!isGPSEnabled && !isNetworkEnabled) {
            } else {
                this.canGetLocation = true;
                if (isNetworkEnabled) {

locationManager.requestLocationUpdates(

LocationManager.NETWORK_PROVIDER,
                            MIN_TIME_BW_UPDATES,

MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
                    if (locationManager != null) {
                        location = locationManager

.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
                        if (location != null) {
                            latitude =
location.getLatitude();
                            longitude =
location.getLongitude();
                        }
                    }
                }
                if (isGPSEnabled) {
                    if (location == null) {

locationManager.requestLocationUpdates(
```

```
LocationManager.GPS_PROVIDER,
                                MIN_TIME_BW_UPDATES,

MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
                        if (locationManager != null) {
                            location =
locationManager.getLastKnownLocation(LocationManager.GPS_
PROVIDER);
                            if (location != null) {
                                latitude =
location.getLatitude();
                                longitude =
location.getLongitude();
                            }
                        }
                    }
                }
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
        return location;
    }
    public void stopUsingGPS(){
        if(locationManager != null){

locationManager.removeUpdates(GPSTracker.this);
        }
    }
    public double getLatitude(){
        if(location != null){
            latitude = location.getLatitude();
        }
        return latitude;
    }
    public double getLongitude(){
        if(location != null){
            longitude = location.getLongitude();
        }
```

```
        return longitude;
    }
    public boolean canGetLocation() {
        return this.canGetLocation;
    }
    public void showSettingsAlert(){
        AlertDialog.Builder alertDialog = new
AlertDialog.Builder(mContext);
        alertDialog.setTitle("GPS is settings");
        alertDialog.setMessage("GPS is not enabled. Do
you want to go to settings menu?");
        alertDialog.setPositiveButton("Settings", new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface
dialog,int which) {
                Intent intent = new
Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                mContext.startActivity(intent);
            }
        });
        alertDialog.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog,
int which) {
                dialog.cancel();
            }
        });
        alertDialog.show();
    }
    @Override
    public void onLocationChanged(Location location) {
        this.location=location;
        latitude=location.getLatitude();
        longitude=location.getLongitude();

view.handlerLocation.sendMessage(view.handlerLocation.obt
ainMessage());
    }
    @Override
    public void onProviderDisabled(String provider) {
    }
```

```
        @Override
        public void onProviderEnabled(String provider) {
        }
        @Override
        public void onStatusChanged(String provider, int
    status, Bundle extras) {
        }
        @Override
        public IBinder onBind(Intent arg0) {
            return null;
        }
    }
```

3. *Source Code class* MainActivity.java

```
package com.DT.netinfota;

import android.annotation.TargetApi;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.os.StrictMode;
import android.telephony.CellLocation;
import android.telephony.NeighboringCellInfo;
import android.telephony.PhoneStateListener;
import android.telephony.ServiceState;
import android.telephony.SignalStrength;
import android.telephony.TelephonyManager;
import android.telephony.gsm.GsmCellLocation;
import android.text.InputType;
import android.util.Log;
import android.view.Gravity;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
```

```
import android.widget.Button;
import android.widget.EditText;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;
import android.widget.Toast;

import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;

class time extends Thread {

    public int waktu = 0;
    private Main_Activity view;
    public time(Main_Activity view) {
        this.view = view;
    }
    public void run() {
        try {
            while (true) {

view.handler.sendMessage(view.handler.obtainMessage());
                waktu++;
                Thread.sleep(1000);
            }
        } catch (Exception e) {
            Log.i("Test", e.getMessage());
        }
    }
}

public class Main_Activity extends Activity implements
View.OnClickListener {
    private static final int INFO_SERVICE_STATE_INDEX = 0;
    private static final int INFO_CELL_LOCATION_INDEX = 1;
    private static final int INFO_CALL_STATE_INDEX = 2;
    private static final int INFO_CONNECTION_STATE_INDEX = 3;
    private String
deviceinfo,call_state,latitude,longitude,serving_time;
    int cellid,lac,kualitas,kuatlevel,scrCd;
    private String
tcall_state=null,tlatitude=null,tlongitude=null,tserving_time=
"000000";
    int tcellid=0123,tlac=0123,tkualitas=0123,tkuatlevel=0123;
    TextView stime;
    Button export_csv,dev_info,call_button,clear_button;
```

```
    TableLayout tLayout;
    private static final int[] info_ids = {
            R.id.serviceState_info,
            R.id.cellLocation_info,
            R.id.callState_info,
            R.id.connectionState_info
    };
    public time waktu;
    public Handler handler = new Handler() {
        @Override
        public void handleMessage(Message msg) {
            stime.setText(waktu.waktu+" s");
        }
    };
    public Handler handlerLocation = new Handler() {
        @Override
        public void handleMessage(Message msg) {

setTextViewText(R.id.viewLong,GPSTracker.longitude+"");

setTextViewText(R.id.viewLat,GPSTracker.latitude+"");
        }
    };
    private GPSTracker gps;
    @TargetApi(Build.VERSION_CODES.GINGERBREAD)
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.dataview_activity);
        StrictMode.enableDefaults();
        startSignalLevelListener();
        displayTelephonyInfo();
        tLayout = (TableLayout) findViewById(R.id.tabel);
        stime = (TextView) findViewById(R.id.serviceTime);
        export_csv = (Button) findViewById(R.id.exportcsv);
        dev_info = (Button) findViewById(R.id.viewdeviceinfo);
        call_button = (Button) findViewById(R.id.callbutton);
        clear_button = (Button)
findViewById(R.id.clearbutton);
        export_csv.setOnClickListener(this);
        dev_info.setOnClickListener(this);
        call_button.setOnClickListener(this);
        clear_button.setOnClickListener(this);
        waktu = new time(this);
        waktu.start();
        view_result();
    }
```

```
    @Override
    protected void onPause() {
        super.onPause();
        //stopListening();
    }

    @Override
    protected void onResume() {
        super.onResume();
        view_result();
        //startSignalLevelListener();
    }

    @Override
    protected void onDestroy() {
        //stopListening();
        //data.nilai.clear();
        super.onDestroy();
    }

    private void setTextViewText(int id, String text) {
        ((TextView) findViewById(id)).setText(text);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.main, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch(item.getItemId()) {
            case R.id.keluar:
                new AlertDialog.Builder(this)
                        .setMessage("Yakin ingin keluar?")
                        .setCancelable(false)
                        .setPositiveButton("Ya", new
DialogInterface.OnClickListener() {
                            public void
onClick(DialogInterface dialog, int id) {
                                data.clear();
                                view_result();
                                waktu.waktu=0;
                                stopListening();
```

```
                                finish();
                                onDestroy();
                            }
                        })
                        .setNegativeButton("Kembali", null)
                        .show();
                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }
    private void update_data() {
        try {
            gps = new GPSTracker(Main_Activity.this,this);
            if(gps.canGetLocation()){
                latitude = ""+gps.getLatitude();
                longitude = ""+gps.getLongitude();
            }else{
                latitude = "GPS Off";
                longitude = "GPS Off";
                //gps.showSettingsAlert();
            }
            setTextViewText(R.id.viewLat,latitude+"");
            setTextViewText(R.id.viewLong,longitude+"");
        } catch (Exception e) {}
        if(call_state!=null && waktu.waktu>0){
            boolean ok=true;
            try{
                if(tcall_state.equals(call_state) &&
                        tlatitude.equals(latitude) &&
                        tlongitude.equals(longitude)){
                    /*if(tcall_state.equals(call_state) &&
                            tcellid==cellid &&
                            tkualitas==kualitas &&
                            tkuatlevel==kuatlevel &&
                            tlac==lac &&
                            tlatitude.equals(latitude) &&
                            tlongitude.equals(longitude)){*/
                        ok=false;
                }
            }catch (Exception e){
            }
            if(ok){
                serving_time=new
SimpleDateFormat("HH:mm:ss").format(new Date());
                try{
                    if(!tserving_time.equals(serving_time)){
```

```
                        addNilai(new
String[]{serving_time,latitude + "", longitude + "",
call_state +"", cellid+ "", lac+ "", kualitas + "", kuatlevel
+ "", scrCd + "", waktu.waktu+""});
                        waktu.waktu = 0;
                    }
                }catch (Exception e){
                }
            }
        }
    }
    private void addNilai(String[] nilai) {
        tcall_state=call_state;
        tcellid=cellid;
        tkuatlevel=kuatlevel;
        tkualitas=kualitas;
        tlac=lac;
        tlatitude=latitude;
        tlongitude=longitude;
        tserving_time=serving_time;
        data.nilai.add(nilai);
        view_result();
    }

    private void view_result() {
        TableLayout.LayoutParams parameterTable = new
TableLayout.LayoutParams(
                TableLayout.LayoutParams.WRAP_CONTENT,
                TableLayout.LayoutParams.WRAP_CONTENT);
        TableRow.LayoutParams parameterRow = new
TableRow.LayoutParams(
                TableRow.LayoutParams.WRAP_CONTENT,
                TableRow.LayoutParams.WRAP_CONTENT);
        tLayout.removeAllViews();
        TableRow tHead = new TableRow(this);
        for (int i = 0; i < data.header.length; i++) {
            TextView val = new TextView(this);
            val.setPadding(7, 1, 7, 1);
            val.setText(data.header[i]);
            val.setTextColor(Color.RED);
            val.setGravity(Gravity.CENTER_HORIZONTAL |
Gravity.CENTER_VERTICAL);
            val.setBackgroundColor(Color.BLACK);
            tHead.setLayoutParams(parameterTable);
            tHead.addView(val, parameterRow);
        }
        tLayout.addView(tHead);
```

```
        int baris=10;
        if(data.nilai.size()<10){
            baris=data.nilai.size();
        }
        for (int x = data.nilai.size()-1; x >=
(data.nilai.size()-baris); x--) {
            TableRow tRow = new TableRow(this);
            for (int i = 0; i < data.header.length; i++) {
                TextView val = new TextView(this);
                val.setPadding(7, 1, 7, 1);
                val.setText(data.nilai.get(x)[i]);
                val.setTextColor(Color.BLACK);
                val.setGravity(Gravity.CENTER_HORIZONTAL |
Gravity.CENTER_VERTICAL);
                if(x%2==0){
                    val.setBackgroundColor(Color.WHITE);
                }else{
                    val.setBackgroundColor(Color.LTGRAY);
                }
                tRow.setLayoutParams(parameterTable);
                tRow.addView(val, parameterRow);
            }
            tLayout.addView(tRow);
        }
    }

    private void stopListening() {
        TelephonyManager tm = (TelephonyManager)
getSystemService(TELEPHONY_SERVICE);
        tm.listen(phoneStateListener,
PhoneStateListener.LISTEN_NONE);
    }

    private void startSignalLevelListener() {
        TelephonyManager tm = (TelephonyManager)
getSystemService(TELEPHONY_SERVICE);
        int events =
PhoneStateListener.LISTEN_SIGNAL_STRENGTHS
                | PhoneStateListener.LISTEN_DATA_ACTIVITY
                | PhoneStateListener.LISTEN_CELL_LOCATION
                | PhoneStateListener.LISTEN_CALL_STATE
                |
PhoneStateListener.LISTEN_CALL_FORWARDING_INDICATOR
                |
PhoneStateListener.LISTEN_DATA_CONNECTION_STATE
                |
PhoneStateListener.LISTEN_MESSAGE_WAITING_INDICATOR
```

```
                | PhoneStateListener.LISTEN_SERVICE_STATE;
        tm.listen(phoneStateListener, events);
    }

    private void displayTelephonyInfo() {
        TelephonyManager tm = (TelephonyManager)
getSystemService(Context.TELEPHONY_SERVICE);
        if (tm.getPhoneType() ==
TelephonyManager.PHONE_TYPE_GSM) {
            final GsmCellLocation loc = (GsmCellLocation)
tm.getCellLocation();
            if (loc != null) {
                cellid = loc.getCid();
                lac = loc.getLac();
                scrCd = loc.getPsc();
            }
        }
        String deviceid = tm.getDeviceId();
        String phonenumber = tm.getLine1Number();
        String softwareversion =
tm.getDeviceSoftwareVersion();
        String operatorname = tm.getNetworkOperatorName();
        String simcountrycode = tm.getSimCountryIso();
        String simoperator = tm.getSimOperatorName();
        String simserialno = tm.getSimSerialNumber();
        String subscriberid = tm.getSubscriberId();
        String networktype =
getNetworkTypeString(tm.getNetworkType());
        String phonetype =
getPhoneTypeString(tm.getPhoneType());
        deviceinfo = "";
        deviceinfo += ("CellID: " + cellid + "\n");
        deviceinfo += ("LAC: " + lac + "\n");
        deviceinfo += ("Device ID: " + deviceid + "\n");
        deviceinfo += ("Phone Number: " + phonenumber + "\n");
        deviceinfo += ("Software Version: " + softwareversion
+ "\n");
        deviceinfo += ("Operator Name: " + operatorname +
"\n");
        deviceinfo += ("SIM Country Code: " + simcountrycode +
"\n");
        deviceinfo += ("SIM Operator: " + simoperator + "\n");
        deviceinfo += ("SIM Serial No.: " + simserialno +
"\n");
        deviceinfo += ("Subscriber ID: " + subscriberid +
"\n");
        deviceinfo += ("Network Type: " + networktype + "\n");
```

```
        deviceinfo += ("Phone Type: " + phonetype + "\n");
        List<NeighboringCellInfo> cellinfo =
tm.getNeighboringCellInfo();
        if (null != cellinfo) {
            for (NeighboringCellInfo info : cellinfo) {
                deviceinfo += ("\tCellID: " + info.getCid() +
", RSSI: " + info.getRssi() + "\n");
            }
        }
        setTextViewText(R.id.cellid, cellid+"");
        setTextViewText(R.id.viewScrCd, scrCd+"");
    }

    private String getNetworkTypeString(int type) {
        String typeString = "Unknown";
        switch (type) {
            case TelephonyManager.NETWORK_TYPE_HSDPA:
                typeString = "HSDPA";
                on3g=true;
                break;
            case TelephonyManager.NETWORK_TYPE_UMTS:
                typeString = "UMTS";
                on3g=true;
                break;
            case TelephonyManager.NETWORK_TYPE_EDGE:
                typeString = "EDGE";
                on3g=false;
                break;
            case TelephonyManager.NETWORK_TYPE_GPRS:
                typeString = "GPRS";
                on3g=false;
                break;
            default:
                typeString = "3G";
                on3g=true;
                break;
        }
        return typeString;
    }

    private String getPhoneTypeString(int type) {
        String typeString = "Unknown";
        switch (type) {
            case TelephonyManager.PHONE_TYPE_GSM:
                typeString = "GSM";
                break;
            default:
```

```
                typeString = "UNKNOWN";
                break;
        }
        return typeString;
    }

    private boolean on3g=false;
    private final PhoneStateListener phoneStateListener = new
PhoneStateListener() {
        @Override
        public void onSignalStrengthsChanged(SignalStrength
signalStrength) {
            displayTelephonyInfo();
            kuatlevel = -
113+2*(signalStrength.getGsmSignalStrength());
            kualitas =
signalStrength.getGsmSignalStrength()/signalStrength.getEvdoDb
m();
            setTextViewText(R.id.viewRxLev, kuatlevel + "
dBm");
            setTextViewText(R.id.viewRxQual, kualitas + "");
            update_data();
            super.onSignalStrengthsChanged(signalStrength);
        }
        @Override
        public void onCallForwardingIndicatorChanged(boolean
cfi) {
            super.onCallForwardingIndicatorChanged(cfi);
        }

        @Override
        public void onCallStateChanged(int state, String
incomingNumber) {
            update_data();
            String callState = "UNKNOWN";
            switch (state) {
                case TelephonyManager.CALL_STATE_IDLE:
                    callState = "IDLE";
                    data.idle++;
                    break;
                case TelephonyManager.CALL_STATE_RINGING:
                    callState = "Ringing";
                    data.ringing++;
                    break;
                case TelephonyManager.CALL_STATE_OFFHOOK:
                    callState = "Calling";
                    data.calling++;
```

```
                    break;
                }
                call_state=callState;
                setTextViewText(info_ids[INFO_CALL_STATE_INDEX],
callState);
                super.onCallStateChanged(state, incomingNumber);
            }

            @Override
            public void onCellLocationChanged(CellLocation
location) {
                String locationString = location.toString();

setTextViewText(info_ids[INFO_CELL_LOCATION_INDEX],
locationString);
                super.onCellLocationChanged(location);
            }
            @Override
            public void onDataConnectionStateChanged(int state) {
                String connectionState;
                switch (state) {
                    case TelephonyManager.DATA_CONNECTED:
                        connectionState = "Connected";
                        break;
                    case TelephonyManager.DATA_CONNECTING:
                        connectionState = "Connecting";
                        break;
                    case TelephonyManager.DATA_DISCONNECTED:
                        connectionState = "Disconnected";
                        break;
                    case TelephonyManager.DATA_SUSPENDED:
                        connectionState = "Suspended";
                        break;
                    default:
                        connectionState = "Unknown: " + state;
                        break;
                }

setTextViewText(info_ids[INFO_CONNECTION_STATE_INDEX],
connectionState);
                super.onDataConnectionStateChanged(state);
            }

            @Override
            public void onMessageWaitingIndicatorChanged(boolean
mwi) {
                super.onMessageWaitingIndicatorChanged(mwi);
```

```
        }

        @Override
        public void onServiceStateChanged(ServiceState
serviceState) {
            String serviceStateString;
            switch (serviceState.getState()) {
                case ServiceState.STATE_IN_SERVICE:
                    serviceStateString = "IN SERVICE";
                    break;
                case ServiceState.STATE_EMERGENCY_ONLY:
                    serviceStateString = "EMERGENCY ONLY";
                    break;
                case ServiceState.STATE_OUT_OF_SERVICE:
                    serviceStateString = "OUT OF SERVICE";
                    break;
                case ServiceState.STATE_POWER_OFF:
                    serviceStateString = "POWER OFF";
                    break;
                default:
                    serviceStateString = "UNKNOWN";
                    break;
            }

setTextViewText(info_ids[INFO_SERVICE_STATE_INDEX],
serviceStateString);
            super.onServiceStateChanged(serviceState);
        }
    };

    @Override
    public void onClick(View view) {
        switch (view.getId()){
            case R.id.exportcsv:
                update_data();
                if(data.write(",")){

Toast.makeText(getApplicationContext(),"Exported to SD
Card/NetInfo",Toast.LENGTH_LONG).show();
                    data.clear();
                    view_result();
                    waktu.waktu=0;
                }else{

Toast.makeText(getApplicationContext(),"Terjadi
Kesalahan",Toast.LENGTH_LONG).show();
                }
```

```
                break;
            case R.id.viewdeviceinfo:
                AlertDialog dialog = new
AlertDialog.Builder(this).setMessage(deviceinfo).show();
                TextView textView = (TextView)
dialog.findViewById(android.R.id.message);
                textView.setTextSize(10);
                break;
            case R.id.callbutton:
                AlertDialog.Builder alert = new
AlertDialog.Builder(this);
                alert.setMessage("Masukkan nomor telepon");
                final EditText input = new EditText(this);
                input.setHint("");

input.setInputType(InputType.TYPE_CLASS_NUMBER);
                alert.setView(input);
                alert.setPositiveButton("Panggil", new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface
dialog, int whichButton) {
                        Intent callIntent = new
Intent(Intent.ACTION_CALL);
                        callIntent.setData(Uri.parse("tel:" +
input.getText()));
                        startActivity(callIntent);
                    }
                });
                alert.show();
                break;
            case R.id.clearbutton:
                new AlertDialog.Builder(this)
                        .setMessage("Hapus semua record
data?")
                        .setCancelable(false)
                        .setPositiveButton("Hapus", new
DialogInterface.OnClickListener() {
                            public void
onClick(DialogInterface dialog, int id) {
                                data.clear();
                                view_result();
                                waktu.waktu = 0;
                            }
                        })
                        .setNegativeButton("Batal", null)
                        .show();
                break;
```

```
        }
    }
}
```

4. Source code dari AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.DT.netinfota"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="16" />
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission
android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission
android:name="android.permission.CALL_PHONE"/>
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.DT.netinfota.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action
android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Lampiran B. Tabel data *drive test* Net Info

1. **Tabel data *drive test* pada 20 Agustus 2013 – 15:33:16 WITA**

No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	Scrambling Code	Serving Time		
1	-8.79745	115.1612	IDLE	52708817	8042	-5	-81	96	27		
2	-8.79745	115.1612	IDLE	52708817	8042	-8	-81	96	34		
3	-8.79785	115.1609	Calling	52708817	8042	-8	-75	96	27		
4	-8.79785	115.1609	Calling	52708817	8042	-9	-75	96	27		
5	-8.79796	115.1609	Calling	52708817	8042	-9	-75	96	15		
6	-8.79799	115.1609	Calling	52708817	8042	-7	-75	96	5	74	Sukses
7	-8.79802	115.1609	IDLE	52708817	8042	-7	-79	96	1		
8	-8.79802	115.1609	IDLE	52708817	8042	-7	-80	96	1		
9	-8.79818	115.1608	IDLE	52708817	8042	-10	-77	96	6		
10	-8.79818	115.1608	IDLE	52708817	8042	-10	-77	96	7		
11	-8.79832	115.1608	IDLE	52708817	8042	-12	-77	96	5		
12	-8.79832	115.1608	IDLE	52708817	8042	-12	-77	96	5		
13	-8.8003	115.1606	Calling	52708817	8042	-10	-77	96	55		
14	-8.8003	115.1606	Calling	52708817	8042	-12	-77	96	10	65	Sukses
15	-8.80033	115.1606	IDLE	52692336	8042	-12	-85	22	2		
16	-8.80033	115.1606	IDLE	52692336	8042	-11	-85	22	1		
17	-8.8006	115.1606	IDLE	52692336	8042	-13	-85	22	9		
18	-8.80062	115.1606	IDLE	52692336	8042	-13	-85	22	10		
19	-8.80254	115.16	Calling	52692336	8042	-12	-85	22	11		
20	-8.80254	115.16	Calling	52692336	8042	-14	-85	22	53	64	Sukses
21	-8.80278	115.16	IDLE	52692336	8042	-14	-85	22	9		
22	-8.80278	115.16	IDLE	52692336	8042	-8	-85	22	8		
23	-8.80299	115.1599	Calling	52692336	8042	-8	-77	22	30		
24	-8.80299	115.1599	Calling	52692336	8042	-8	-77	22	22	62	Sukses

25	-8.80307	115.1599	IDLE	52692336	8042	-13	-70	22	20		
26	-8.80307	115.1599	IDLE	52692336	8042	-13	-65	22	3		
27	-8.80459	115.1594	Calling	52692336	8042	-7	-70	22	8		
28	-8.80462	115.1594	Calling	52692336	8042	-7	-70	22	58	66	Sukses
29	-8.80464	115.1594	IDLE	52692336	8042	-5	-70	22	1		
30	-8.80464	115.1594	IDLE	52692336	8042	-5	-67	22	1		
31	-8.80495	115.1593	IDLE	52692336	8042	-6	-67	22	12		
32	-8.80498	115.1593	IDLE	52692336	8042	-6	-67	22	13		
33	-8.80513	115.1593	Calling	52563933	8011	-5	-67	11	5		
34	-8.80513	115.1593	Calling	52563933	8011	-5	-67	11	5		
35	-8.80521	115.1592	Calling	52563933	8011	-7	-65	11	44		
36	-8.80521	115.1592	Calling	52563933	8011	-10	-70	11	10	64	Sukses
37	-8.8059	115.159	IDLE	52563933	8042	-11	-75	11	24		
38	-8.8059	115.159	IDLE	52563933	8042	-11	-78	11	10		
39	-8.80681	115.1588	Calling	52563933	8042	-8	-70	11	12		
40	-8.80681	115.1588	Calling	52563933	8042	-8	-65	11	15		
41	-8.80737	115.1586	Calling	52563933	8042	-8	-67	11	18		
42	-8.80737	115.1586	Calling	52563933	8042	-8	-67	11	18	63	Sukses
43	-8.80758	115.1585	IDLE	52692336	8042	-4	-70	22	8		
44	-8.80758	115.1585	IDLE	52692336	8042	-4	-70	22	8		
45	-8.80766	115.1585	IDLE	52708817	8042	-5	-72	96	3		
46	-8.80766	115.1585	IDLE	52708817	8042	-5	-72	96	3		
47	-8.80774	115.1585	IDLE	52563933	8042	-9	-68	11	3		
48	-8.80774	115.1585	IDLE	52563933	8042	-9	-68	11	3		
49	-8.80785	115.1584	IDLE	52563933	8042	-5	-70	11	5		
50	-8.80785	115.1584	IDLE	52563933	8042	-6	-85	11	5		
51	-8.80789	115.1584	IDLE	52563933	8042	-7	-85	11	14		
52	-8.80789	115.1584	IDLE	52563933	8011	-8	-85	11	14		
53	-8.80804	115.1584	IDLE	52563933	8011	-8	-80	11	35		
54	-8.80804	115.1584	IDLE	52563933	8011	-16	-80	11	35		

55	-8.80811	115.1583	IDLE	52563933	8011	-16	-97	11	2		
56	-8.80811	115.1583	Calling	52563933	8011	-17	-105	11	6	6	Block Call
57	-8.80832	115.1583	IDLE	52563933	8011	-25	-105	11	9		
58	-8.80832	115.1583	IDLE	52563933	8011	-22	-91	11	9		
59	-8.80841	115.1583	IDLE	52563933	8011	-11	-91	11	5		
60	-8.80843	115.1583	IDLE	52563933	8011	-11	-91	11	5		
61	-8.80868	115.1582	Calling	52563933	8011	-5	-85	11	10		
62	-8.80868	115.1582	Calling	52563933	8011	-5	-85	11	10		
63	-8.81038	115.1576	Calling	52563933	8011	-6	-85	11	51	71	Sukses
64	-8.8104	115.1576	IDLE	52563933	8011	-6	-85	11	10		
65	-8.81048	115.1576	IDLE	52563933	8011	-6	-79	11	3		
66	-8.81048	115.1576	IDLE	52563933	8011	-6	-79	11	3		
67	-8.81065	115.1576	IDLE	52563933	8011	-8	-69	11	6		
68	-8.81065	115.1576	IDLE	52563933	8011	-7	-69	11	6		
69	-8.81071	115.1576	IDLE	52563933	8011	-7	-69	11	2		
70	-8.81071	115.1576	IDLE	52563933	8011	-7	-69	11	2		
71	-8.81328	115.1572	Calling	52563933	8011	-9	-69	11	71		
72	-8.81328	115.1572	Calling	52563933	8011	-6	-69	11	2	73	Sukses
73	-8.81354	115.1571	IDLE	52563933	8011	-6	-77	11	7		
74	-8.81354	115.1571	IDLE	52563933	8011	-7	-77	11	7		
75	-8.81369	115.1571	IDLE	52563933	8011	-7	-77	11	6		
76	-8.81369	115.1571	IDLE	52563933	8011	-6	-77	11	6		
77	-8.81398	115.157	Calling	52563933	8011	-5	-71	11	56		
78	-8.81398	115.157	Calling	52563933	8011	-6	-71	11	9	65	Sukses
79	-8.81537	115.1563	IDLE	52563933	8011	-6	-77	11	20		
80	-8.81537	115.1563	IDLE	52563933	8011	-6	-77	11	22		
81	-8.81574	115.156	Calling	52563933	8011	-7	-77	11	17		
82	-8.81574	115.156	Calling	52563933	8011	-7	-77	11	50	67	Sukses
83	-8.81576	115.156	IDLE	52563933	8011	-8	-60	11	2		

84	-8.81576	115.156	IDLE	52563933	8011	-7	-60	11	2		
85	-8.81612	115.1556	IDLE	52563933	8011	-7	-83	11	13		
86	-8.81616	115.1556	IDLE	52563933	8011	-8	-83	11	14		
87	-8.8162	115.1556	Calling	52563935	8011	-4	-71	27	10		
88	-8.8162	115.1556	Calling	52563935	8011	-4	-71	27	26		
89	-8.81684	115.155	Calling	52563935	8011	-8	-79	27	25	61	Sukses
90	-8.81684	115.155	IDLE	52563935	8011	-13	-79	27	24		
91	-8.81825	115.1541	Calling	52563935	8011	-13	-79	27	30		
92	-8.81825	115.1541	Calling	52563935	8011	-13	-79	27	46	76	Sukses
93	-8.81836	115.154	IDLE	52563935	8011	-14	-90	27	3		
94	-8.81836	115.154	IDLE	52563935	8011	-14	-90	27	2		
95	-8.81854	115.1539	IDLE	52563935	8011	-14	-94	27	7		
96	-8.81854	115.1539	IDLE	52563935	8011	-14	-93	27	7		
97	-8.81991	115.1519	Calling	52563935	8011	-14	-90	27	62		
98	-8.81991	115.1519	Calling	52563935	8011	-14	-90	27	5	67	Sukses
99	-8.81994	115.1518	IDLE	52563935	8011	-8	-91	27	1		
100	-8.81994	115.1518	IDLE	52563935	8011	-8	-91	27	2		
101	-8.82008	115.1514	IDLE	52563935	8011	-8	-90	27	13		
102	-8.82009	115.1514	IDLE	52563935	8011	-10	-92	27	14		
103	-8.82067	115.1489	Calling	52563935	8011	-12	-90	27	55		
104	-8.82067	115.1489	Calling	52563935	8011	-12	-94	27	12	67	Sukses
105	-8.82085	115.1478	IDLE	52563935	8011	-12	-94	27	33		
106	-8.82085	115.1478	IDLE	52563935	8011	-12	-93	27	33		
107	-8.82091	115.1474	IDLE	52563935	8011	-12	-93	27	9		
108	-8.82091	115.1474	IDLE	52563935	8011	-14	-91	27	9		
109	-8.82101	115.1469	Calling	52563935	8011	-13	-92	27	15		
110	-8.82101	115.1469	Calling	52563935	8011	-14	-93	27	14		
111	-8.82158	115.1448	Calling	52563935	8011	-10	-94	27	52		
112	-8.8216	115.1447	Calling	52563935	8011	-10	-94	27	10	91	Sukses
113	-8.82228	115.1436	IDLE	52574943	8011	-11	-91	299	34		

114	-8.82228	115.1436	IDLE	52574943	8011	-11	-91	299	34		

Ringing : 0

Calling : 17

2. Tabel data *drive test* pada 20 Agustus 2013 – 16:21:44 WITA

No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	Scrambling code	Serving Time		
1	-8.82267	115.1433	IDLE	52574943	8011	-13	-90	299	7		
2	-8.82269	115.1433	Calling	52574943	8011	-12	-90	299	1		
3	-8.82434	115.1418	Calling	52574943	8011	-12	-90	299	42		
4	-8.82508	115.1411	Calling	52574943	8011	-13	-92	299	17	68	Sukses
5	-8.82544	115.1409	Calling	52574943	8011	-14	-90	299	8		
6	-8.82605	115.1404	IDLE	52574943	8011	-10	-92	299	15		
7	-8.82609	115.1404	Calling	52574945	8011	-11	-92	299	1		
8	-8.82673	115.14	Calling	52574945	8011	-11	-92	299	14		
9	-8.82835	115.1387	Calling	52574945	8011	-12	-90	299	36	69	Sukses
10	-8.8289	115.1378	Calling	52574945	8011	-12	-90	299	18		
11	-8.82928	115.1371	IDLE	52574945	8011	-12	-90	299	17		
12	-8.82929	115.137	Calling	52571984	8011	-12	-94	56	1		
13	-8.82949	115.1365	Calling	52571984	8011	-11	-94	56	12		
14	-8.83068	115.1359	Calling	52571984	8011	-12	-89	56	39		
15	-8.8312	115.1358	Calling	52571984	8011	-12	-90	56	14	70	Sukses
16	-8.83141	115.1358	Calling	52571984	8011	-10	-90	56	4		
17	-8.83149	115.1358	IDLE	52571984	8011	-6	-90	56	2		
18	-8.83154	115.1358	IDLE	52571985	8011	-10	-97	72	10		
19	-8.83155	115.1358	IDLE	52571985	8011	-14	-85	72	76		

Ringing : 0

Calling : 3

3. Tabel data *drive test* pada 20 Agustus 2013 – 16:26:00 WITA

No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	Scrambling Code	Serving Time		
1	-8.83155	115.1358	IDLE	52571985	8011	-14	-83	72	6		
2	-8.83171	115.1357	IDLE	52571985	8011	-13	-83	72	11	68	Sukses
3	-8.83174	115.1357	Calling	52571985	8011	-12	-89	72	1		
4	-8.83362	115.1366	Calling	52571985	8011	-12	-89	72	67		
5	-8.83373	115.1365	IDLE	52571985	8011	-9	-105	72	2		
6	-8.83388	115.1365	IDLE	52571985	8011	-8	-97	72	3	70	Sukses
7	-8.83404	115.1365	IDLE	52574944	8011	-9	-105	307	3		
8	-8.83725	115.1368	Calling	52574944	8011	-9	-105	307	70		
9	-8.8373	115.1368	IDLE	52574944	8011	-15	-99	307	1		
10	-8.83744	115.1369	IDLE	52574944	8011	-8	-105	307	2		
11	-8.83774	115.1371	IDLE	52574944	8011	-8	-105	307	8	68	Sukses
12	-8.83959	115.1382	Calling	52574944	8011	-7	-99	307	60		
13	-8.83968	115.1385	Calling	52574944	8011	-7	-99	307	8		
14	-8.83973	115.1386	IDLE	52574944	8011	-7	-99	307	9		

Ringing : 0

Calling : 3

4. Tabel data *drive test* pada 20 Agustus 2013 – 16:43:42 WITA

No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	Scrambling Code	Serving Time		
1	-8.83975	115.1387	IDLE	52574944	8011	-10	-90	307	3		
2	-8.83975	115.1387	IDLE	52574944	8011	-10	-90	307	14		
3	-8.84002	115.1399	Calling	52574944	8011	-7	-99	307	33	70	Sukses
4	-8.84139	115.1406	Calling	52574944	8011	-7	-99	307	37		
5	-8.84164	115.1405	IDLE	52705817	8042	-10	-100	98	5		
6	-8.84199	115.1404	IDLE	52705817	8042	-10	-101	98	5		
7	-8.84229	115.1404	Calling	52570013	8011	-9	-99	298	6		
8	-8.84299	115.1411	Calling	52570013	8011	-9	-98	298	22		
9	-8.84301	115.1418	Calling	52570013	8042	-10	-93	298	16		
10	-8.84227	115.1419	Calling	52570013	8042	-7	-99	298	14	68	Sukses
11	-8.84197	115.1421	Calling	52570013	8042	-7	-99	298	10		
12	-8.84191	115.1422	IDLE	52705815	8042	-10	-67	306	8		
13	-8.8421	115.142	IDLE	52705815	8042	-6	-67	306	17		
14	-8.84239	115.1418	IDLE	52705817	8042	-5	-65	98	8		
15	-8.84266	115.1418	IDLE	52705817	8042	-7	-99	98	6		
16	-8.84281	115.1419	IDLE	52705817	8042	-11	-91	98	9		
17	-8.84281	115.142	IDLE	52705817	8042	-7	-99	98	3		
18	-8.84283	115.1421	IDLE	52705817	8042	-6	-99	98	6		
19	-8.84363	115.1417	IDLE	52705817	8042	-15	-98	98	90		
20	-8.84363	115.1418	IDLE	52705817	8042	-12	-97	98	2		
21	-8.84341	115.1433	Calling	52705817	8042	-8	-97	98	42	69	Sukses
22	-8.84288	115.1442	Calling	52705817	8042	-8	-97	98	27		
23	-8.84289	115.1443	IDLE	52705817	8042	-13	-113	98	3		
24	-8.8429	115.1447	IDLE	52705817	8042	-10	-107	98	6		
25	-8.84295	115.1451	Calling	52705817	8042	-10	-101	98	9		
26	-8.843	115.1454	Calling	52705817	8011	-9	-95	98	6		

27	-8.84303	115.146	Calling	52705817	8011	-12	-89	98	12	66	Sukses
28	-8.84435	115.1469	Calling	52705817	8011	-12	-89	98	39		
29	-8.84445	115.147	IDLE	52705817	8011	-10	-99	98	3		
30	-8.84451	115.1471	IDLE	52705817	8011	-10	-93	98	3		
31	-8.84468	115.1472	IDLE	52705817	8011	-10	-93	98	5		
32	-8.84597	115.1484	Calling	52705817	8042	-13	-87	98	48	62	Sukses
33	-8.84599	115.1485	Calling	52705817	8042	-13	-87	98	14		
34	-8.84589	115.1484	IDLE	52705817	8042	-13	-87	98	28		

Ringing : 0

Calling : 5

5. Tabel data *drive test* pada 20 Agustus 2013 – 16:57:39 WITA

No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	Scrambling Code	Serving Time		
1	-8.84296	115.1464	IDLE	52577754	8011	-9	-95	90	5		
2	-8.84254	115.1464	Calling	52577754	8011	-12	-89	90	18		
3	-8.84264	115.1475	Calling	52577754	8042	-9	-95	90	24	64	Sukses
4	-8.84282	115.1486	Calling	52577754	8042	-9	-95	90	22		
5	-8.84219	115.149	IDLE	52574944	8011	-8	-105	307	18		
6	-8.84187	115.1492	IDLE	52574944	8011	-8	-105	307	8		
7	-8.84172	115.1492	Calling	52570013	8011	-8	-97	298	4		
8	-8.84037	115.1511	Calling	52570013	8042	-11	-91	298	40		
9	-8.8406	115.1518	Calling	52570013	8042	-8	-97	298	14	64	Sukses
10	-8.84067	115.152	Calling	52570013	8042	-8	-97	298	6		
11	-8.84071	115.1521	IDLE	52705815	8042	-14	-90	306	4		
12	-8.84014	115.1526	IDLE	52705815	8042	-14	-75	306	17		

13	-8.84007	115.1526	Calling	52705815	8042	-13	-87	306	1	68	Sukses
14	-8.83772	115.1526	Calling	52705815	8042	-13	-87	306	67		
15	-8.83762	115.1522	IDLE	52705815	8042	-14	-90	306	7		
16	-8.83749	115.1522	IDLE	52705815	8042	-11	-91	306	3		
17	-8.8371	115.1522	IDLE	52577754	8011	-14	-90	90	6		
18	-8.83702	115.152	IDLE	52577754	8011	-14	-90	90	7		
19	-8.83694	115.1519	Calling	52577754	8011	-13	-100	90	2		
20	-8.83584	115.1519	Calling	52577754	8011	-14	-98	90	20		
21	-8.83541	115.1505	Calling	52577754	8011	-14	-90	90	38	69	Sukses
22	-8.8353	115.1502	Calling	52577754	8011	-12	-89	90	9		
23	-8.83531	115.1501	IDLE	52577754	8042	-15	-83	90	1		
24	-8.83522	115.1498	IDLE	52577754	8011	-11	-91	90	8		
25	-8.83521	115.1497	IDLE	52577754	8011	-11	-91	90	2		
26	-8.8342	115.1494	Calling	52577754	8011	-14	-85	90	22		
27	-8.83308	115.1496	Calling	52577754	8011	-5	-77	90	20		
28	-8.83239	115.1496	Calling	52577754	8011	-6	-71	90	11	66	Sukses
29	-8.83158	115.1495	Calling	52577754	8011	-5	-71	90	13		
30	-8.83146	115.1495	IDLE	52577754	8011	-5	-63	90	1		
31	-8.83099	115.1496	IDLE	52577754	8011	-6	-63	90	9		
32	-8.83057	115.1497	Calling	52577755	8011	-10	-69	98	6	62	Sukses
33	-8.82762	115.1497	Calling	52577755	8011	-10	-69	98	56		
34	-8.82756	115.1497	IDLE	52577755	8011	-7	-75	98	1		
35	-8.82721	115.15	IDLE	52577753	8011	-7	-81	82	5		
36	-8.82696	115.1501	IDLE	52577753	8011	-8	-81	82	6		
37	-8.82581	115.151	Calling	52577753	8011	-6	-87	82	24		
38	-8.82529	115.1514	Calling	52577753	8011	-10	-93	82	12		
39	-8.82383	115.1518	Calling	52577753	8011	-12	-81	82	29	68	Sukses
40	-8.82361	115.1519	Calling	52577753	8011	-14	-81	82	3		
41	-8.82329	115.1521	IDLE	52577753	8011	-13	-87	82	7		
42	-8.82284	115.1524	IDLE	52577753	8011	-12	-87	82	11	62	Sukses

43	-8.81944	115.1535	Calling	52577753	8011	-13	-87	82	62		
44	-8.81941	115.1536	IDLE	52563935	8011	-13	-81	27	5		
45	-8.81941	115.1536	IDLE	52563935	8011	-14	-81	27	1		

Ringing : 0

Calling : 8

6. Tabel data *drive test* pada 20 Agustus 2013 – 17:26:48 WITA

No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	Scrambling Code	Serving Time		
1	-8.81588	115.1562	IDLE	52563935	8011	-14	-83	27	9		
2	-8.81588	115.1562	IDLE	52563935	8011	-14	-83	27	2	78	Sukses
3	-8.81704	115.1573	Calling	52563935	8011	-13	-83	27	78		
4	-8.81712	115.1573	IDLE	52563935	8011	-13	-83	27	70		
5	-8.81714	115.1574	Calling	52563934	8011	-13	-75	82	1		
6	-8.81841	115.1591	Calling	52563934	8011	-14	-81	82	44		
7	-8.81905	115.16	Calling	52563934	8011	-14	-75	82	23	70	Sukses
8	-8.8192	115.1602	Calling	52563934	8011	-9	-75	82	2		
9	-8.81924	115.1603	IDLE	52563934	8011	-6	-73	82	1		
10	-8.81946	115.1605	IDLE	52563934	8011	-6	-73	82	6		
11	-8.82074	115.1616	Calling	52563934	8011	-5	-79	82	29		
12	-8.82131	115.162	Calling	52563934	8011	-14	-73	82	12		
13	-8.82175	115.1623	Calling	52563934	8011	-12	-79	82	8	68	Sukses
14	-8.82258	115.1628	Calling	52563934	8011	-12	-78	82	19		
15	-8.82268	115.1629	IDLE	52563934	8011	-6	-83	82	1		
16	-8.82298	115.1632	IDLE	52563934	8011	-6	-80	82	8		
17	-8.82319	115.1633	Calling	52572525	8011	-14	-78	102	4		

18	-8.82357	115.1637	Calling	52572525	8011	-14	-80	102	9	68	Sukses
19	-8.82591	115.1659	Calling	52572525	8011	-10	-79	102	55		
20	-8.82599	115.1659	IDLE	52572524	8011	-9	-77	94	1		
21	-8.82612	115.166	IDLE	52572524	8011	-5	-87	94	3		
22	-8.82643	115.166	IDLE	52572524	8011	-7	-79	94	6		
23	-8.82658	115.1659	IDLE	52572524	8011	-7	-79	94	3		
24	-8.82796	115.1668	Calling	52572524	8011	-7	-79	94	53	67	Sukses
25	-8.82836	115.1665	Calling	52572524	8011	-10	-79	94	14		
26	-8.82821	115.1658	IDLE	52572524	8011	-10	-79	94	16		
27	-8.82987	115.1653	Calling	52561243	8011	-10	-73	9	36	70	Sukses
28	-8.83089	115.1641	Calling	52561243	8011	-6	-73	9	34		
29	-8.83108	115.164	IDLE	52561243	8011	-9	-83	9	4		
30	-8.83146	115.1638	IDLE	52561243	8011	-9	-83	9	8		
31	-8.83151	115.1638	Calling	52561243	8011	-7	-73	9	1		
32	-8.83237	115.1636	Calling	52561243	8011	-8	-79	9	15		
33	-8.83278	115.1637	Calling	52561243	8011	-5	-80	9	6		
34	-8.83316	115.163	Calling	52561243	8011	-5	-75	9	21		
35	-8.83393	115.1626	Calling	52561243	8011	-8	-75	9	24	70	Sukses
36	-8.83399	115.1625	Calling	52561243	8011	-8	-76	9	3		
37	-8.83407	115.1624	IDLE	52561243	8011	-5	-77	9	3		
38	-8.83422	115.1621	IDLE	52561243	8011	-5	-77	9	7		
39	-8.83443	115.1616	Calling	52561243	8011	-8	-77	9	11		
40	-8.83383	115.1605	Calling	52561243	8011	-6	-65	9	30		
41	-8.83362	115.1596	Calling	52561243	8042	-5	-67	9	18		
42	-8.83382	115.1591	Calling	52561243	8011	-5	-77	9	9	70	Sukses
43	-8.83378	115.159	Calling	52561243	8011	-5	-77	9	2		
44	-8.83375	115.1589	IDLE	52577754	8011	-6	-87	90	1		
45	-8.83361	115.1586	IDLE	52577754	8011	-4	-77	90	6		
46	-8.83337	115.158	IDLE	52577754	8011	-5	-77	90	13		
47	-8.83428	115.1575	Calling	52577754	8011	-6	-85	90	23	69	Sukses

48	-8.83572	115.1559	Calling	52577754	8011	-5	-85	90	46		
49	-8.8365	115.1553	IDLE	52577754	8011	-10	-77	90	17		
50	-8.837	115.1549	IDLE	52577754	8011	-10	-77	90	10		
51	-8.83791	115.1544	Calling	52577754	8011	-8	-83	90	23		
52	-8.83776	115.1534	Calling	52577754	8042	-8	-89	90	18		
53	-8.8376	115.1522	Calling	52577754	8042	-6	-83	90	21	70	Sukses
54	-8.83726	115.1522	Calling	52577754	8042	-5	-83	90	8		
55	-8.8371	115.1521	IDLE	52577754	8011	-6	-77	90	4		
56	-8.83693	115.1518	IDLE	52577754	8011	-6	-77	90	7		

Ringing : 0

Calling : 10

7. Tabel data *drive test* pada 21 Agustus 2013 – 10:09:37 WITA

No	Latitude	Longitude	Call State	Cell ID	LAC	Ec/No	RSCP	Scrambling Code	Serving Time		
1	-8.84276	115.1403	IDLE	52570013	8011	-5	-67	298	2		
2	-8.84276	115.1403	IDLE	52570013	8011	-5	-67	298	10		
3	-8.84276	115.1403	IDLE	52570013	8011	-7	-67	298	6		
4	-8.84276	115.1403	IDLE	52570013	8011	-10	-67	298	7		
5	-8.84323	115.1397	Calling	52570013	8011	-10	-73	298	15		
6	-8.84326	115.1397	Calling	52570013	8011	-9	-73	298	35	66	Sukses
7	-8.84376	115.1397	Calling	52570013	8011	-7	-81	298	16		
8	-8.84376	115.1397	IDLE	52570013	8011	-5	-81	298	16		
9	-8.84445	115.1404	IDLE	52570013	8011	-7	-87	298	44		

10	-8.84445	115.1404	Calling	52570013	8011	-7	-87	298	27		
11	-8.84445	115.1404	Calling	52570013	8011	-8	-87	298	17		
12	-8.84445	115.1404	Calling	52570013	8011	-7	-87	298	19	63	Sukses
13	-8.84444	115.1402	IDLE	52570013	8011	-7	-87	298	4		
14	-8.84444	115.1402	IDLE	52570013	8011	-7	-87	298	5		
15	-8.84421	115.1396	Calling	52570014	8011	-8	-81	306	21		
16	-8.84418	115.1396	Calling	52570014	8011	-8	-81	306	20		
17	-8.84295	115.1394	Calling	52570014	8011	-6	-81	306	2	67	Sukses
18	-8.84295	115.1394	Calling	52570014	8011	-7	-81	306	24		
19	-8.84298	115.1393	IDLE	52570014	8011	-10	-75	306	4		
20	-8.84299	115.1393	IDLE	52570014	8011	-10	-75	306	3		
21	-8.84308	115.1389	IDLE	52570014	8011	-10	-75	306	58		
22	-8.84309	115.1389	IDLE	52570014	8011	-11	-75	306	58		
23	-8.84314	115.1387	Calling	52570013	8011	-12	-81	298	8		
24	-8.84315	115.1386	Calling	52570013	8011	-7	-81	298	8		
25	-8.84359	115.1386	Calling	52570013	8011	-6	-89	298	15		
26	-8.84363	115.1386	Calling	52570013	8011	-6	-89	298	15		
27	-8.84404	115.1383	Calling	52570013	8011	-9	-95	298	36		
28	-8.84403	115.1383	Calling	52570013	8011	-9	-95	298	36		
29	-8.844	115.1382	Calling	52570013	8011	-9	-95	298	6	130	Sukses
30	-8.844	115.1382	Calling	52570013	8011	-9	-95	298	6		
31	-8.844	115.1382	IDLE	52570013	8011	-6	-95	298	3		
32	-8.844	115.1382	IDLE	52570013	8011	-6	-90	298	2		
33	-8.844	115.1382	IDLE	52570013	8011	-6	-90	298	5		

Ringing : 0

Calling : 4