GYSURV : APLIKASI *MOBILE SURVEILLANCE SYSTEM* BERBASISKAN TEKNOLOGI GSM DAN UMTS

SKRIPSI

Oleh

M. GINTA MARDALIN

040403710X

DEPARTEMEN TEKNIK ELEKTRO

FAKULTAS TEKNIK UNIVERSITAS INDONESIA

GENAP 2007/2008

GYSURV : APLIKASI *MOBILE SURVEILLANCE SYSTEM* BERBASISKAN TEKNOLOGI GSM DAN UMTS

SKRIPSI

Oleh

M. GINTA MARDALIN

040403710X

SKRIPSI INI DIAJUKAN UNTUK MELENGKAPI SEBAGIAN PERSYARATAN MENJADI SARJANA TEKNIK

DEPARTEMEN TEKNIK ELEKTRO

FAKULTAS TEKNIK UNIVERSITAS INDONESIA

GENAP 2007/2008

PERNYATAAN KEASLIAN SKRIPSI

Saya menyatakan dengan sesungguhnya bahwa skripsi dengan judul :

***GYSURV* : APLIKASI *MOBILE SURVEILLANCE SYSTEM* BERBASISKAN TEKNOLOGI GSM DAN UMTS**

yang dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada program studi Teknik Elektro Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia, sejauh yang saya ketahui bukan merupakan tiruan atau duplikasi dari skripsi yang sudah dipublikasikan dan atau pernah dipakai untuk mendapatkan gelar kesarjanaan di lingkungan Universitas Indonesia maupun di Perguruan Tinggi atau instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

Depok, 26 Februari 2008

(M. Ginta Mardalin)

NPM 04 04 03 710 X

PENGESAHAN

Skripsi dengan judul :

***GYSURV* : APLIKASI *MOBILE SURVEILLANCE SYSTEM* BERBASISKAN TEKNOLOGI GSM DAN UMTS**

dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada program studi Teknik Elektro Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia dan telah disidangkan pada tanggal 26 februari 2008.

Depok, 26 Februari 2008

Dosen Pembimbing,

Muhammad Suryanegara, ST. MSc.

NIK 040 705 018 9

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada :

Muhammad Suryanegara, ST. MSc.

selaku Dosen Pembimbing skripsi yang telah bersedia meluangkan waktunya untuk memberikan bimbingan, petunjuk, dan saran-saran, serta kepada :

Prof.Dr.Ir.Dadang Gunawan, MEng

selaku Ketua *Wireless* dan *Signal Processing Work Group* yang telah membiayai skripsi ini sepenuhnya, sehingga skripsi ini dapat diselesaikan dengan baik.

M.Ginta Mardalin
NPM 04 04 03 710 X
Departemen Teknik Elektro

Dosen Pembimbing
Muhammad Suryanegara, ST, MSc.

GYSURV : APLIKASI _MOBILE SURVEILLANCE SYSTEM_ BERBASISKAN TEKNOLOGI GSM DAN UMTS

ABSTRAK

Skripsi ini membangun *GySurv*, sebuah aplikasi *mobile surveillance system*, dengan memanfaatkan teknologi selular GSM dan UMTS.

Pada *mobile surveillance system* ini awalnya keadaan suatu ruangan ditangkap menggunakan kamera, lalu hasil keluaran dari kamera ini dikirimkan pada suatu *broadcaster* yang akan melakukan proses *encoding* dan pembentukan *file* SDP. Video yang telah di*encoding* dan *file-file* SDP ini lalu dikirimkan ke *streaming server* yang akan melayani pengiriman *file* ini kepada *client* melalui jaringan internet. Transmisi, baik dari *broadcaster* ke *streaming server*, maupun dari *streaming server* ke *client* dilakukan dengan melibatkan protokol pada lapisan aplikasi, lapisan trans*port*, dan lapisan jaringan.

Untuk dapat mengambil *stream data* yang dikirimkan *streaming server*, maka *client* membutuhkan sebuah perangkat lunak. Perangkat lunak ini bangun dengan menggunakan platform J2ME.

Ujicoba dan analisa dilakukan dengan mengukur *delay* yang terjadi, serta mengganti parameter *data rate* dan *frame rate* yang mempengaruhinya. Dari hasil ujicoba diketahui bahwa *delay* rata-rata yang terjadi saat sistem diterapkan pada jaringan GSM adalah 23,652 detik, sedangakan *delay* rata-rata yang terjadi apabila sistem diterapkan pada jaringan UMTS adalah 9,681 detik.

Kata kunci : GSM, UMTS, *Streaming*, J2ME, *Data rate, Frame rate*.

M.Ginta Mardalin
NPM 04 04 03 710 X
Departemen Teknik Elektro

Counsellor
Muhammad Suryanegara, ST, MSc.

GYSURV : *MOBILE SURVEILLANCE SYSTEM APPLICATION* BASED ON GSM AND UMTS TECHNOLOGY

ABSTRACT

This Project build GySurv, a mobile surveillance system application which is based on GSM and UMTS cellular technology.

This mobile surveillance system is preceded by capturing a room condition by camera, then the output of the camera is transmitted to the *broadcaster*, where the files is encoded and the SDP file is builded. The encoded video and the SDP files are transmitted to the streaming server, which will send it to the clients via internet. The transmission from *broadcaster* to streaming server and transmission from streaming server to clients, use some protokols from application layer, trans*port* layer, and network layer.

In order to take data stream, that being send by streaming server, the clients need a capturing *software*. This *software* is made by using java programming language in J2ME platform.

The trial and analysis is done by measuring delay that happens, and by controlling the *data* rate and frame rate. The trial show that this system average delay, when it's using GSM networks is 23,652 seconds, while the system average delay, when it's using UMTS networks is 9,681 seconds

Keywords : GSM, UMTS, Streaming, J2ME, Encode rate, Frame rate.

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR SINGKATAN

3G	Third Generation
API	Application Programming *Interface*
CDC	Connected Device Configuration
CLDC	Connected LimitedDevice Configuration
DMZ	De-Militarized Zone
IP	Internet Protocol
J2EE	Java 2 Enterprise Edition
J2ME	Java 2 Micro Edition
J2SE	Java 2 Second Edition
JNI	Java Native *Interface*
JSR	Java Specification Request
LTE	Long Term Evolution
MIDP	*Mobile* Device Pro*file*
RA	Real Audio
RM	Rreal Media
RTCP	Real Time Control Protocol
RTP	Real Time Protocol
RTSP	Real Time *Streaming* Protocol
SDP	Session Description Protocol
SMIL	Synchronized Media Integration Language
TCP	Transfer Control Protocol
UDP	Unit Datagram Protocol
VOD	Video On Demand
VOIP	Vioce Over IP

DAFTAR ISTILAH

Circuit Switching	Salah satu teknik switching yang menerapkan sebuah path komunikasi yang *dedicated* (permanen) antara 2 buah station.
Packet Switching	Salah satu teknit switching dimana data yang ditransmisikan dibagi-bagi ke dalam paket-paket kecil dan hubungan *single node-to-node* dapat di*share* secara dinamis oleh banyak paket.
Flow control	Salah satu fungsi kontrol pada TCP untuk menyesuaikan laju pengiriman data dengan keadaan jaringan untuk menghindari *congestion*.
Error control	Salah satu fungsi kontrol pada TCP yang menjamin sampainya data yang dikirimkan.
Congestion control	Salah satu fungsi kontrol pada TCP yang mendeteksi adanya congestion pada jaringan.

BAB I
PENDAHULUAN

1.1. LATAR BELAKANG MASALAH

Pada saat ini, teknologi informasi sudah berkembang dengan pesat. Salah satu contohnya adalah teknologi seluler, yang berkembang mulai dari generasi pertama sampai generasi ketiga pada saat ini. Pada teknologi generasi pertama, informasi yang dapat dikirimkan hanyalah berupa suara dengan menggunakan *circuit switching*. Lalu lahirlah teknologi GSM, dimana informasi yang berupa data juga dapat dikirimkan dengan *circuit switching* [1]. Hal ini menandakan kelahiran teknologi generasi kedua. Pada teknologi selular generasi ketiga, pengiriman informasi baik berupa suara maupun data sudah dapat dilakukan dengan *packet switching.* Selain itu, sistem generasi ketiga ini didukung dengan *bit rate* mencapai 2 Mbps [2]. Hal ini memungkinkan banyak aplikasi yang dapat dilakukan pada teknologi generasi ketiga (3G) ini, seperti *video call, real time video sharing, online multiplayer games, VOIP (Video Over IP), browsing* internet, *content download, mobile tv,* dan aplikasi multimedia lainnya.

Skripsi ini membahas rancang bangun *GySurv*, aplikasi *mobile surveillance system* berbasis GSM dan UMTS. Dengan aplikasi *GySurv*, sebuah ponsel dapat digunakan sebagai perangkat suatu sistem keamanan (*surveillance*), sehingga pengguna sistem keamanan dapat memantau dan mengendalikan sistem keamanan tersebut secara terpusat dengan mobilitas yang tinggi.

Dengan menggunakan layanan audio dan video *streaming* pada ponsel, situasi pada suatu ruangan dapat dipantau. Awalnya keadaan ruangan ditangkap dengan kamera *surveillance*, lalu data berupa video dan suara ini dikirimkan ke *streaming server*, untuk kemudian dikirimkan ke ponsel. Selain itu, dengan didukungnya packet switching pada teknologi GPRS, maka dapat ditambahkan

berbagai *value added service* pada sistem ini, seperti pengiriman hasil *snapshot* video *surveillance* sebagai MMS.

Agar ponsel dapat digunakan sebagai salah satu perangkat pada suatu sistem keamanan, maka ponsel ini memerlukan suatu perangkat lunak yang digunakan sebagai antarmuka yang dapat membantu pengguna dalam memantau dan mengendalikan suatu sistem keamanan. Perangkat lunak tambahan ini dapat dibuat menggunakan bahasa JAVA yang merupakan bahasa pemprogaraman *portabel*, yang dapat dijalankan pada perangkat *standalone* dengan berbagai sistem operasi.

1.2. PERUMUSAN MASALAH

Konsep sistem keamanan diimplementasikan pada suatu ruangan. Pemantauan ruangan dilakukan menggunakan ponsel berbasiskan jaringan UMTS dan GSM.

1.3. TUJUAN

1. Membangun aplikasi *Mobile Surveillance* System berbasiskan jaringan GSM dan UMTS.
2. Mengukur dan menganalisa unjuk kerja *Mobile Surveillance System*.
3. Mengatur parameter-parameter pada *Mobile Surveillance* System, sehingga dihasilkan sebuah service dengan biaya murah dan performa yang baik.

1.4. BATASAN MASALAH

Pada skripsi ini, masalah dibatasi pada rancang bangun dan analisa sebuah *Mobile Surveillance* System. Rancang bangun ini meliputi pembuatan suatu sistem *real time streaming* audio dan video, perancangan perangkat lunak pada ponsel sebagai antarmuka antara pengguna dan sistem keamanan yang akan dibangun. Analisa dan ujicoba *Mobile Surveillance* System dilakukan secara real time, kecuali untuk fitur-fitur tambahan yang dilakukan secara *offline* (*localhost*).

1.5. METODOLOGI PENELITIAN

Pada skripsi ini, penelitian dilakukan dengan membangun sebuah *Mobile Surveillance* System yang kemudian diatur parameter-parameter penunjangnya, untuk mendapatkan performa system yang optimal dengan biaya yang minimal. Rancang bangun dan analisa *Mobile Surveillance* ini dilakukan berdasarkan referensi-referensi.

1.6. SISTEMATIKA PENULISAN

Skripsi ini terdiri dari 5 bab dimana sistematika penulisan yang diterapkan dalam skripsi ini menggunakan urutan sebagai berikut :

BAB I PENDAHULUAN

Membahas tentang latar belakang pemilihan tema, perumusan masalah, tujuan, batasan masalah, dan sistematika penulisan.

BAB II LANDASAN TEORI

Membahas tentang teori *real time streaming* audio dan video, yang akan diimplementasikan dalam pentransferan data berupa video dan audio dari kamera *surveillance* ke ponsel. Lalu dibahas juga mengenai teori dasar bahasa pemprograman J2ME yang akan digunakan untuk merancang perangkat lunak pada ponsel.

BAB III PERANCANGAN *MOBILE SURVEILLANCE* SISTEM

Membahas tentang perancangan *Mobile surveillance system* yang terdiri dari kamera *surveillance*, *streaming server*, dan ponsel. Selain itu, dilakukan juga perancangan perangkat lunak yang mencakup diagram alir dan algoritma dari setiap subsistem perangkat lunak yang akan dibuat.

BAB IV ANALISA DAN UJICOBA

Membahas mengenai hasil pengujian dan evaluasi unjuk kerja dari *Mobile surveillance system* yang dijalankan pada jaringan GSM dan UMTS.

BAB V KESIMPULAN

BAB II
LIVE VIDEO STREAMING

2.1. *LIVE STREAMING*

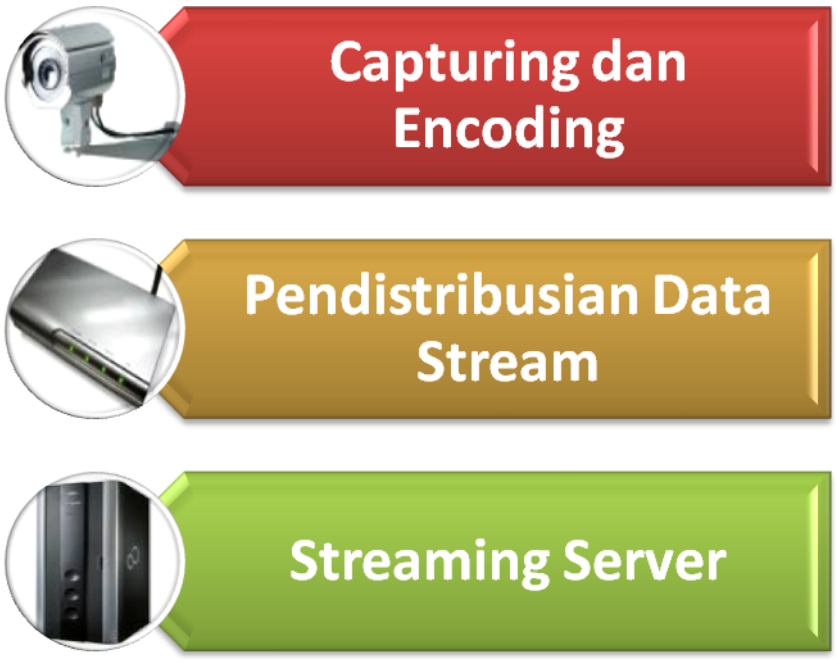
Teori ini diambil dari landasan teori pada seminar ”Rancang Bangun Mobile Surveillance System” oleh M.Ginta Mardalin.

Pengiriman data pada suatu jaringan dapat dilakukan dengan beberapa cara. Cara yang umumnya dilakukan adalah pengunduhan (*downloading*) dan *streaming*. Pada proses pengunduhan data pada jaringan *client-server*, awalnya *client* meminta data yang akan diunduh kepada *server*, lalu *server* mencari data tersebut pada *database*nya untuk kemudian dikirimkan kepada *client* dengan menggunakan beberapa protokol. Pada proses pengunduhan ini, data yang diminta *client* harus diterima seluruhnya, sebelum dapat di decoding dan digunakan. Setelah data ini diterima seluruhnya, maka data dapat digunakan seperti data lokal oleh *client*. Kekurangan utama dalam proses pengunduhan ini adalah dibutuhkannya waktu dalam pengiriman kesuluruhaan data sebelum dapat digunakan, terutama untuk data-data dengan ukuran yang besar, seperti video. Selain itu, proses pengunduhan ini hanya dapat dilakukan pada data yang bersifat diskrit, sedangkan untuk data yang bersifat kontinu, seperti video, akan dihasilkan data dengan kualitas yang rendah.

Sama halnya dengan proses pengunduhan, proses *streaming* diawali oleh permintaan suatu data oleh *client*. Perbedaan utama antara proses *streaming* dengan proses pengunduhan adalah data dapat digunakan walaupun seluruh data belum diterima. Misalnya pada proses *streaming* video, video dapat dimainkan bersamaan dengan pengiriman data yang belum selesai. Agar proses ini dapat dilakukan, maka data yang akan di-*streaming* harus dapat dipecah menjadi bagian-bagian yang lebih kecil yang dapat digunakan secara independen, contohnya adalah pembentukan *frame -frame* dari sebuh video. Selain itu, setiap *frame* ini harus sampai pada *client* sebelum dimainkan. Jadi, setelah *client* meminta *server* untuk mengirimkan video dengan proses *streaming*, *server* akan

mengirimkan video dalam bentuk *frame -frame*. Setelah *frame* pertama diterima, *client* akan langsung memainkan *frame* ini sambil menerima *frame* kedua, hal ini terjadi terus menerus sampai *frame* terakhir dikirimkan. Proses *streaming* ini sesuai dengan konsep *pipelening* data. Penerapan konsep ini akan mengurangi *delay* yang terjadi bila beberapa *client* meminta data yang sama pada sebuah *server*.

Streaming data berupa video dapat dibagi menjadi dua kategori, yaitu *video on demand* (VOD) dan *live streaming* (*live broadcast*). VOD merupakan proses video *streaming* dimana video yang akan di-*streaming* sudah terlebih dahulu direkam dan disimpan pada sebuah *streaming server*. Sedangkan pada *live broadcast*, video yang ditangkap langsung di-*streaming* secara *real-time*, tanpa harus disimpan terlebih dahulu. Pada *live broadcast*, *streaming server* hanya berfungsi untuk menyampaikan video yang akan di-*streaming*. Untuk melakukan video *streaming* terdapat beberapa hal penting yang perlu diperhatikan, seperti yang ditunjukan pada gambar 2.1.



Gambar 2.1. Tahapan dalam proses video *streaming*

2.1.1. Capturing dan Encoding

Pada tahap ini, video yang diambil dalam bentuk analog diubah menjadi digital, misalnya *Audio-Video Interleaved* (.avi), sehingga bisa diproses dengan komputer secara digital. Lalu *file* ini dikompres dan dikurangi *data rate*nya agar sesuai dengan *bandwidth* yang tersedia untuk proses *streaming*. Untuk mengkompres *file* ini digunakan *codec*, yaitu sebuah *software* yang menggunakan algoritma matematis untuk kompresi-dekompres. *Codec* ini digunakan juga pada sisi pengguna (*player*) untuk mengembalikan data *streaming* hasil kompresi menjadi bentuk audio/video semula. *File-file* hasil kompresi ini biasanya memiliki format .mpg, .mp4 atau .rm. Agar *file-file* ini dapat dikirimkan dengan *real-time streaming*, maka perlu ditambahkan metadata dan informasi mengenai kontrol waktu yang akan digunakan oleh *streaming server* untuk mengatur kecepatan pengiriman *file-file* ini. Sebelum *file-file* ini didistribusikan kepada pengguna yang meminta, *file-file* ini harus dibentuk menjadi paket-paket IP, sehingga dapat ditransmisikan melalui jaringan yang berbasis IP (internet).

2.1.2. Pendistribusian Data *Streaming*

Pentransmisian melalui jaringan internet melibatkan beberapa protokol pada lapisan trans*port* dan lapisan aplikasi. Video *streaming* bisa dilakukan menggunakan *Hyper-Text Transfer Protocol* (HTTP) yang bekerja berdasarkan *Transmission Control Protocol* (TCP). TCP adalah protokol yang *Connectionless-oriented* yang mendukung pengiriman data dengan reliabilitas yang tinggi, karena segmen-segmen pada TCP sudah dilengkasi dengan *flow control, error control*, dan *congestion control* [3]. Kelebihan video *streaming* dengan TCP adalah sebagai berikut :

- Tidak diperlukan tambahan *server* sebagai *streaming server*, karena proses *streaming* dapat dilakukan oleh *web server*.
- Dapat melalui berbagai *firewall* dan *gateway*, karena video *streaming* dianggap sama dengan konten *web* biasa.
- Didukung oleh banyak media *player*.

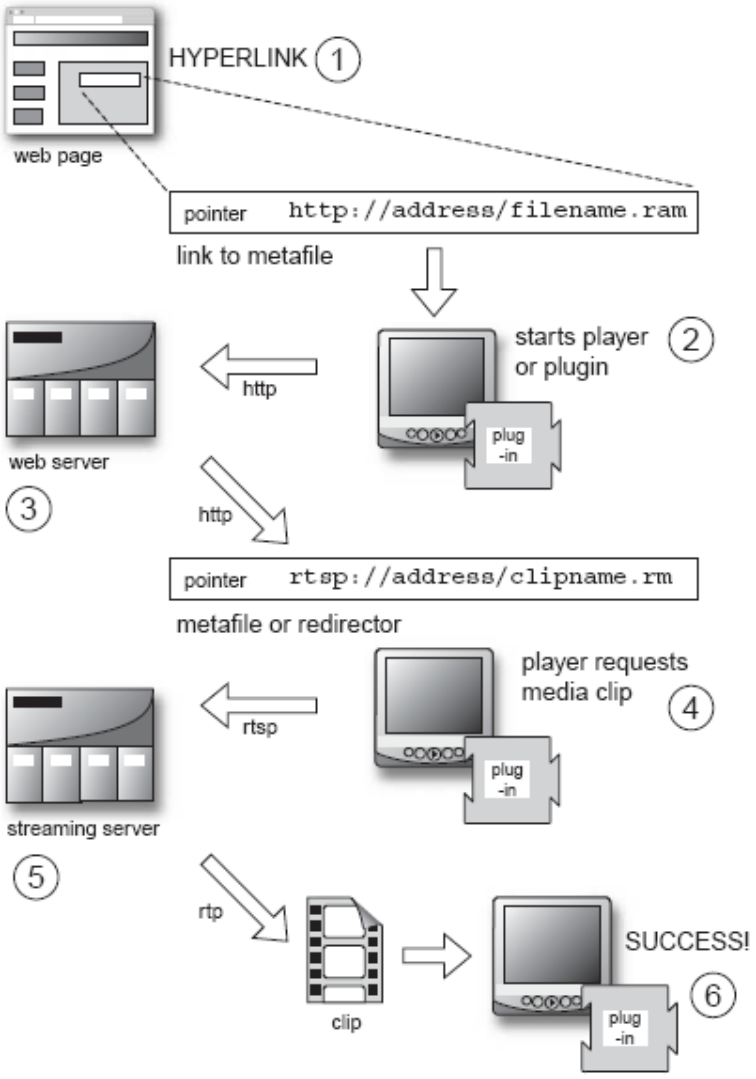
Akan tetapi dibalik semua kelebihan-kelebihan ini, terdapat kekurangan yang sangat signifikan dalam penggunaan TCP pada video *streaming*, yaitu performanya yang buruk. Hal ini dikarenakan TCP dirancang untuk mendukung pengiriman data yang membutuhkan reliabilitas yang tinggi, seperti email. *Error control* pada TCP berfungsi untuk memastikan segmen-segmen sampai pada penerima dalam urutan yang tepat. *Error control* juga memerlukan sebuah ACK (*acknowledgment*), untuk menghentikan pengiriman sebuah segmen ke penerima. Selain itu, pada TCP hilangnya sebuah segmen akan diartikan sebagai tanda bahwa jaringan sedang macet, sehingga *congestion control* akan mengurangi *window size* pada pengiriman berikutnya. Semua perangkat kontrol yang mendukung reliabilitas pada TCP ini akan menimbulkan *delay* pada pengiriman segmen-segmen. *Delay* ini akan membuat segmen yang dikirimkan menjadi sia-sia. Karena pada video *streaming*, yang penting adalah setiap segmen yang dikirimkan harus sampai pada waktu tertentu agar dihasilkan video yang baik.

Selain menggunakan TCP, video *streaming* juga dapat menggunakan *User Datagram Protocol* (UDP) pada lapisan trans*port*. UDP adalah protokol yang tidak mendukung reliabilitas, sehingga dengan menggunakan UDP, tidak akan terjadi *delay* pada pengiriman datagram, karena ptotokol ini tidak didukung dengan *error control*, *flow control* dan *congestion control*. Sehingga dapat disimpulkan bahwa protokol yang lebih sesuai untuk video *streaming* adalah UDP, karena akan menghasilkan video *streaming* dengan performa yang lebih baik.

Akan tetapi, dengan menggunakan UDP tidak berarti reliabilitas tidak penting. Untuk mendukung reliabilitas maka pada lapisan aplikasi dapat digunakan beberapa protokol yang mendukung pengiriman data dengan *streaming*, seperti *Real-Time Streaming Protocol* (RTSP), *Real-time Transport Protocol* (RTP), dan RTP *Control Protocol* (RTCP).

RTSP adalah protokol pada lapisan aplikasi yang didesain untuk mengatur pengiriman data berupa media yang memiliki informasi pewaktuan seperti audio dan video [3]. Karena RTSP adalah protokol pada lapisan aplikasi, maka penggunaaan RTSP tidak tergantung pada penggunaan protokol pada lapisan trans*port*. Sehingga RTSP bisa digunakan berpasangan, baik dengan TCP maupun

UDP. RTSP biasanya digunakan pada jaringan *peer to peer* (dua arah), dimana setiap informasi yang dikirimkan diatur oleh masing-masing *host*. Selain itu, RTSP merupakan protokol yang digunakan hanya untuk mengatur pengiriman data, seperti *remote control*, bukan untuk mengirimkan data. Pada video *streaming*, biasanya RTSP digunakan pada proses pengontrolan video, seperti *pause, rewind* dan lain-lain, sedangkan pengiriman videonya sendiri biasanya dilakukan melalui RTP. Pada gambar 2.2 ditunjukan langkah-langkah dalam pengiriman data dengan RTSP.



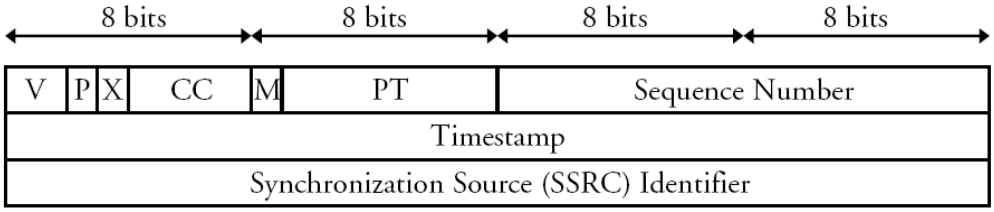
Gambar 2.2. Video *streaming* melalui RTSP [4]

Dalam pengaturan pengiriman data yang akan di-*streaming*, RTSP akan mengirimkan sebuah *Session Description* protokol (SDP) kepada *web browser*. SDP mendeskripsikan satu atau lebih *file* presentasi, dimana masing-masing *file* presentasi berkaitan dengan sebuah media *streaming*. Dalam contoh pada gambar diatas SDP yang yang digunakan adalah .ram. *File* SDP ini bertipe MIME dan memiliki alamat dan nama *file* dari konten yang akan di-*streaming*, yang lengkap. SDP juga berisi informasi mengenai data yang akan di-*streaming* seperti format dan *codec* yang digunakan. Setelah *file* SDP diterima oleh *player*, maka *player* akan langsung meminta konten *streaming* kepada *streaming server* melalui RTSP.

RTP adalah protokol yang didesain untuk mengirimkan data pada aplikasi *real-time*, seperti audio dan video *streaming*. Sama seperti RTSP, penggunaan RTP tidak terkait dengan protokol lapisan dibawahnya, sehingga RTP dapat digunaka berpasangan baik dengan TCP maupun UDP. Akan tetapi, RTP sering digunakan berpasangan dengan UDP. RTP mendukung pengiriman data secara *unicast* dan *multicast*. RTCP adalah protokol pasangan dari RTP yang berfungsi mengontrol kualitas dari media yang dikirimkan melalui RTP. RTCP juga menyediakan beberapa fungsi kontrol pada proses sinkronisasi, *monitoring*, dan juga kontrol-kontrol lainnya seperti *delay*.

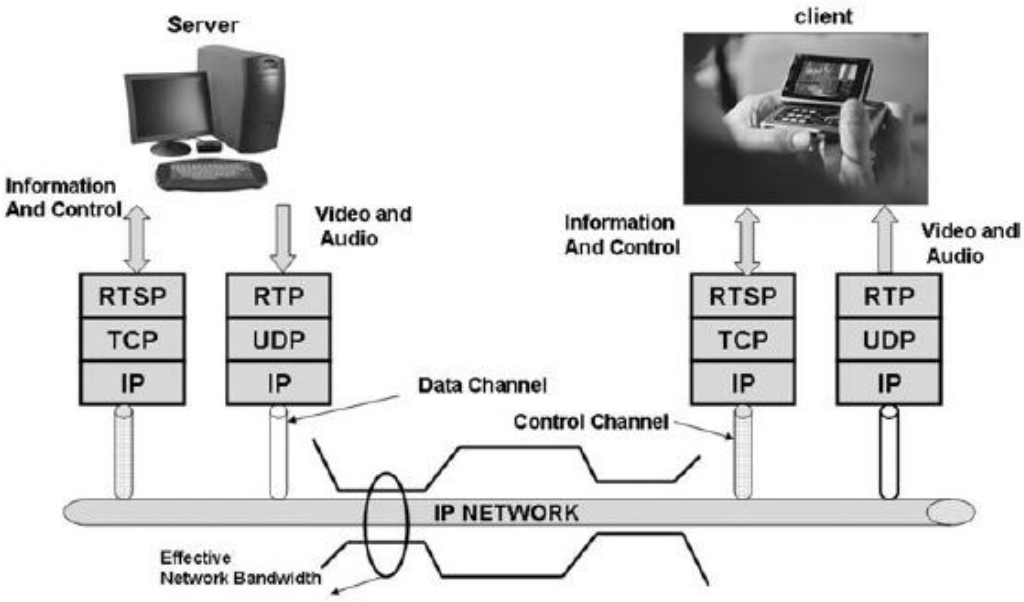
Berbeda dengan UDP dan TCP yang merupakan protokol yang berdiri sendiri pada lapisan trans*port*, RTP/RTCP terintegrasi dalam aplikasi yang menggunakannya. RTP/RTCP tidak menyediakan mekanisme QOSnya sendiri, melainkan hanya menyediakan beberapa umpan balik dari kualitas dan beberapa *frame work* yang dibutuhkan oleh *developer* untuk membuat mekanisme QOSnya sendiri, yang sesuai dengan aplikasi dan jaringan yang digunakan.

Header RTP terdiri dari 12 bit yang tetap, yang dimiliki oleh setiap paket RTP, dan 20 bit tambahan seperti yang ditunjukan pada gambar 2.3. Setiap paket RTP memiliki *timestamp* dan *sequence number* yang berfungsi mengurutkan setiap paket yang sampai pada penerima. Selain itu setiap RTP paket juga memiliki tipe *payload* (PT) yang berfungsi untuk menginformasikan pengirim (dalam hal ini *streaming server*) untuk mengadaptasikan *data rate codec* sesuai dengan *bandwidth* yang tersedia pada jaringan pada saat pengiriman media. Hal ini menjadikan RTP/RTCP protokol yang fleksibel.



Gambar 2.3. *Header* pada RTP [3]

Setiap paket RTP hanya dapat digunakan untuk mengirimkan satu media *streaming*. Misalnya pada pengiriman MP4, maka audio dan video akan dikirimkan menggunakan paket yang berbeda. *Timestamp* dari paket video dan audio ini dapat berbeda, karena *timestamp* tidak diambil secara *real-time* melainkan sesuai dengan pencuplikan waktu pada *clock*, yang tergantung dari tipe *payload* dan aplikasinya. Hal ini memungkinkan antara video dan audio tidak sinkron. Oleh karena itu, digunakan sebuah paket untuk mensinkronisasi keduanya, paket ini dikirimkan menggunakan RTCP. Pensitribusian video *streaming* melalui protocol-protokol ini diilustrasikan pada gambar 2.4.



Gambar 2.4. Pendistibusian video *streaming* [5]

Selain itu, RTCP dapat mengirimkan umpan balik mengenai kualitas pengiriman data dari *player* kepada *server* pengirim. Umpan balik ini dapat berupa laporan mengenai paket-paket yang hilang, atau paket-paket yang diterima tidak berurutan. setelah laporan-laporan ini sampai, *streaming server* akan

meresponnya, misalnya dengan memperkecil ukuran *frame* video atau dengan mengurangi *data rate* pada proses *encoding*. Jadi, dapat diambil kesimpulan bahwa dalam video *streaming* RTSP/TCP digunakan untuk mengirimkan data-data kontrol, sedangkan RTP/UDP digunakan untuk mengirimkan videonya, seperti pada gambar 2.4.

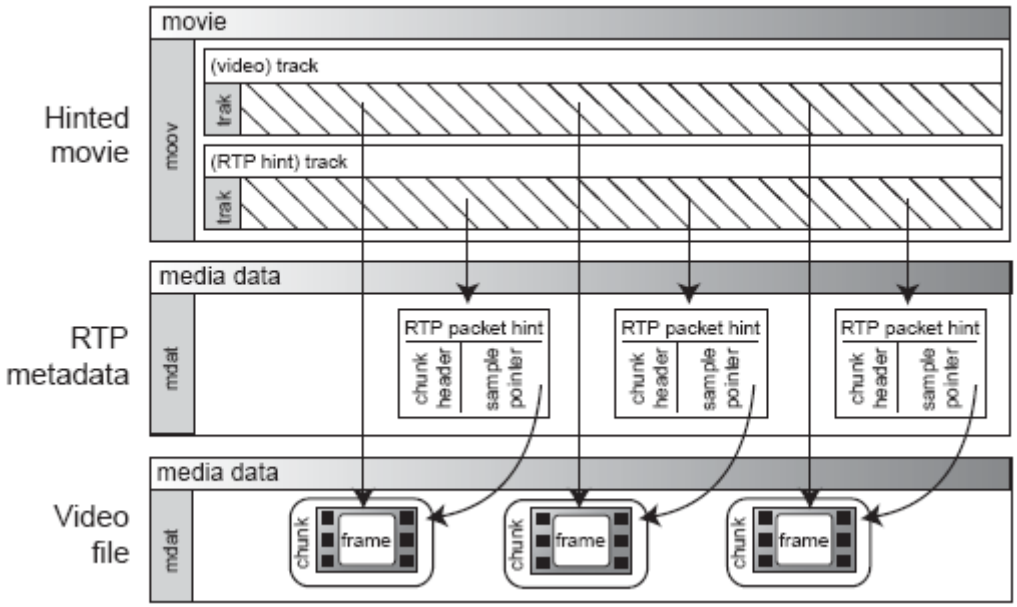
2.1.3. *Streaming Server*

Setelah video/audio di *encoding*, maka video dikirim ke *server*. *Server* yang dapat digunakan adalah *streaming server* atau *web server* yang telah tersedia. Kedua *server* ini memiliki beberapa kelebihan dan kekurangan dalam pengunaannya untuk proses *streaming*, seperti yang ditunjukan pada tabel 2.1.

Tabel 2.1. Perbandingan *web server* dan *streaming server* [4]

	Web Server	*Streaming Server*
Keuntungan	• Tidak memerlukan perangkat tambahan. • Tidak memerlukan keahlian tambahan	• Pengiriman media yang optimal • *Streaming* kontrol yang dinamis • Media kontrol yang interaktif • Mendukung *multicast* • Mendukung *live web*casting • Didukung oleh hardware yang cocok untuk *streaming*
Kekurangan	Hanya mendukung dowload yang bersifat progresif	Membutuhkan perangkat dan keahlian tambahan

Streaming server berfungsi untuk melayani permintaan data-data yang akan di-*streaming*. Sebelum data (video) ini dapat di-*streaming*, video ini harus diubah formatnya menjadi *file* yang cocok untuk proses *streaming*, misalnya MPEG-4, RM, RA. Format-format ini memiliki informasi mengenai kontrol waktu yang akan digunakan *server* untuk mengatur laju pengiriman. Salah satu bentuk format *file* untuk proses *streaming* ditunjukan pada gambar 2.5.

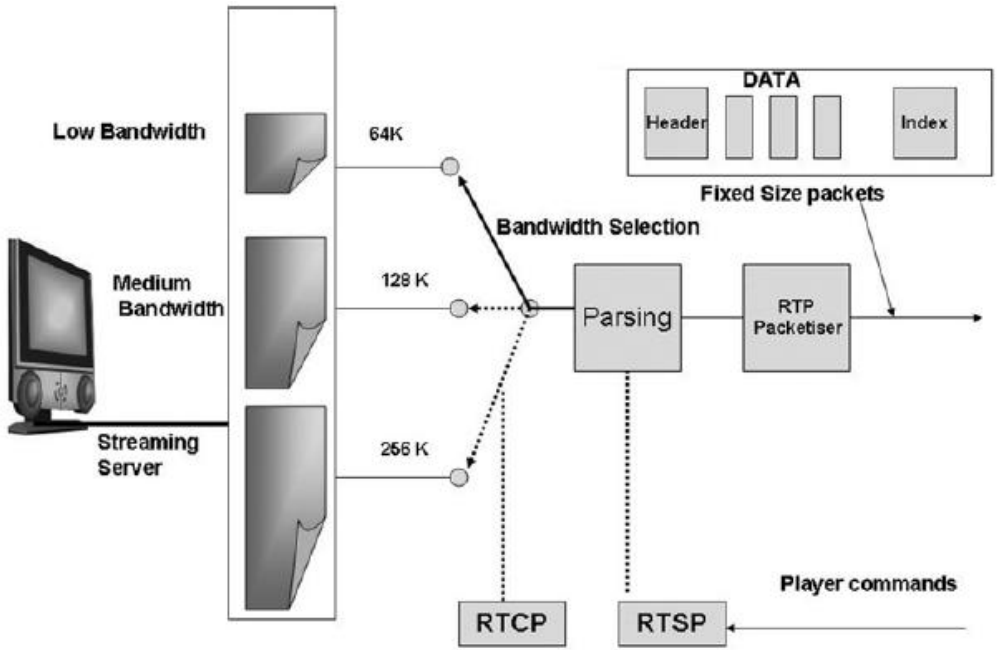


Gambar 2.5. *Streaming* format dan *hint track* [4]

File ini sering disebut *movie*. Sebuah *movie* dapat terdiri dari beberapa media *streaming* seperti audio, video atau media lainnya. Akan tetapi, *movie* tidak hanya terdiri dari audio dan video saja, ia juga memiliki beberapa instruksi untuk menampilkan media data ini. Sebuah *movie* yang telah di-*streaming* akan memiliki tambahan komponen, yaitu *hint track*. *Hint track* ini berfungsi untuk memberikan informasi kepada *server* untuk mengirimkan media yang sesuai melalui RTP. Selain itu, informasi ini juga digunakan oleh *server* untuk mengirimkan video dengan urutan yang benar dan laju yang sesuai. Setelah *file* ini terbentuk, video *streaming* sudah siap untuk dilakukan.

Hal lain yang dilakukan oleh *streaming server* adalah menyesuaikan laju pengiriman data dengan keadaan jaringan seperti yang ditunjukan pada gambar 2.6. Untuk jaringan dengan kapasitas yang besar, maka *server* akan mengirimkan video dengan *data rate* yang tinggi. Akan tetapi bila jaringannya penuh maka laju pengiriman video harus dikurangi. Untuk melakukan hal ini *server* dapat menggunakan laporan dari RTCP untuk mengetahui keadaan jaringan dan menyesuaikannya dengan laju pengiriman video. Selain itu, hal ini juga didukung oleh fasilitas *surestream* pada *broadcaster* yang memungkinkan beberapa *file* yang di-*encoding* dengan *data rate* yang berbeda-beda digabungkan menjadi satu

file. *Streaming server* akan memilih *file* yang akan dikirimkan sesuai dengan keadaan jaringan.

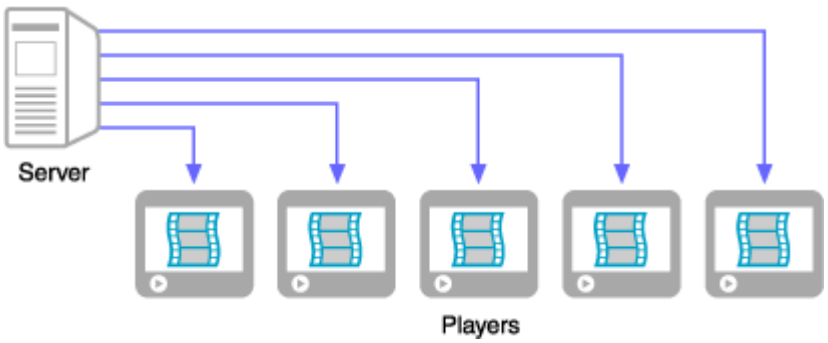


Gambar 2.6. *Surestream* [5]

Pengiriman video melalui *streaming* dapat dilakukan dengan beberapa cara, yaitu :

2.1.3.1. Unicast

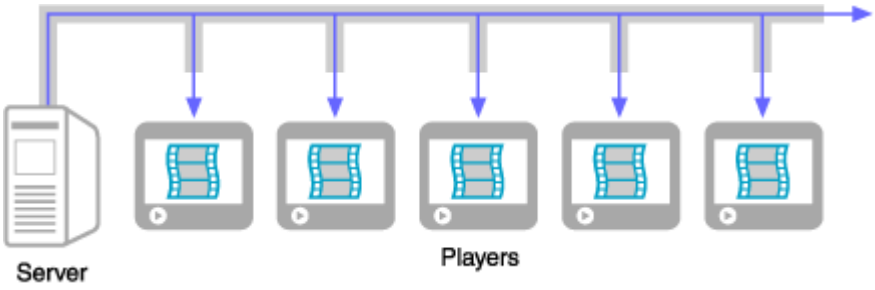
Unicast adalah cara pengiriman yang paling sederhana, dimana *server* mengirimkan video *streaming* yang berbeda untuk setiap media player, seperti yang ditunjukan pada gambar 2.7. Setiap video *streaming* ini akan menggunakan *bandwidth* yang sama, sehingga semakin banyak video *streaming* yang dikirimkan maka semakin besar *bandwidth* yang digunakan.



Gambar 2.7. *Unicast* [6]

2.1.3.2. Multicast

Apabila video *streaming* dikirimkan kepada banyak player dengan *unicast*, maka akan dibutuhkan banyak *bandwidth*. Untuk mengatasi masalah ini, dapat digunakan cara kedua, yaitu *multicast.* Pada *multicast*, *server* hanya mengirimkan satu *streaming* untuk beberapa player, sedangkan untuk mendistrisbusikan *streaming* ke masing-masing player dilakukan oleh *router*, hal ini ditunjukan pada gambar 2.8. Untuk melakukan cara ini dibutuhkan konfigurasi terhadap *router*nya. Router yang digunakan harus mendukung *multicast*. Kebutuhan untuk merubah konfigurasi *router* menyebabkan *multicast* lebih cocok dilakukan untuk jaringan intranet daripada jaringan internet. Selain itu, *server* juga tidak bisa beradaptasi dengan keadaan jaringan secara langsung, karena *server* hanya mengirimkan satu *streaming* kepada banyak player melalui jaringan dengan keadaan yang berbeda-beda.



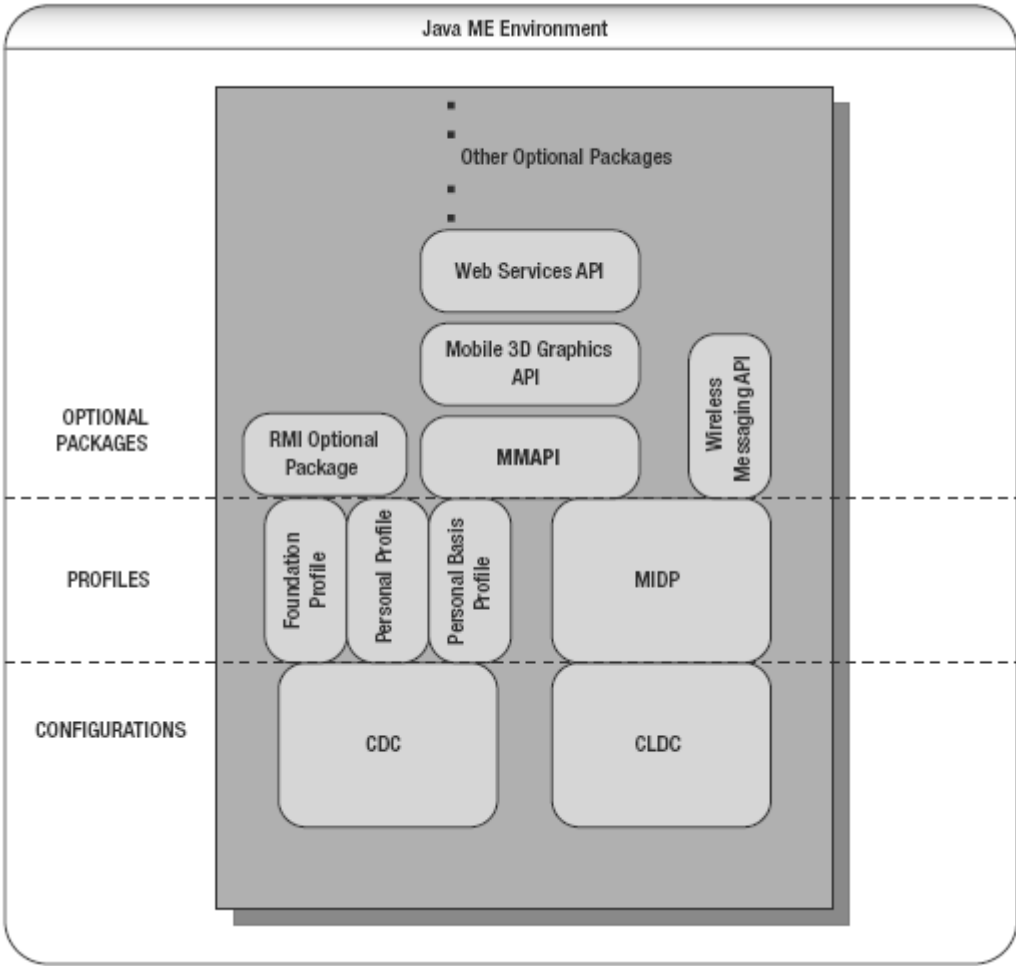
Gambar 2.8. *Multicast* [6]

2.2. J2ME

Teori ini diambil dari landasan teori pada seminar ”Rancang Bangun Mobile Surveillance System” oleh M.Ginta Mardalin.

J2ME adalah salah satu *platform* java yang digunakan untuk membuat perangkat lunak pada peralatan-peralatan yang bersifat perfasif yang memiliki keterbatasan memori dan sumber daya lainnya, seperti pager, ponsel dan PDA. Gambar 2.9 menunjukan arsitektur J2ME. J2ME ditargetkan untuk peralatan perfasif yang memiliki sumber daya yang beraneka ragam, sehingga J2ME

didesain terdiri dari beberapa spesifikasi yang membentuk sebuah *platform*. Spesifikasi-spesifkasi yang digunakan untuk membentuk *platform* pada suatu peralatan perfasif tergantung dari sumber daya yang dimiliki peralatan tersebut. J2ME terdiri dari *configuration* dan *profile*, serta beberapa API seperti yang ditunjukan pada gambar 2.9.



Gambar 2.9. Arsitektur J2ME [8]

2.2.1. *Configuration*

Configuration pada J2ME berfungsi untuk menspesifikasikan fitur dari bahasa pemprograman java dan *java virtual machine* yang akan digunakan, serta menentukan library dan API (*Application Programming Interface*) yang akan digunakan. Pada J2ME terdapat 2 buah konfigurasi, yaitu CDC (*Connected Device Configuration*) dan CLDC (*Connected, Limited Device Configuration*).

2.2.1.1. CDC (Connected Device Configuration)

CDC adalah *configuration* yang digunakan pada peralatan *high-end* yang memiliki memori lebih dari 2 Mb baik RAM maupun ROM seperti internet TV dan *smart phone*. Selain itu peralatan yang mendukung CDC memiliki prosesor yang lebih baik daripada peralatan *low-end* yang menggunakan CLDC. Pada CDC digunakan CVM (*C-Virtual Machine*) sebagai *interpreter*nya. CVM adalah *interpreter* yang menyerupai JVM (*Java Virtual Machine*) pada J2SE, perbedaannya hanyalah pada letak algoritma *garbage collection* pada *virtual machine*. Pada CVM algoritma *garbage collection* dipisahkan dari *virtual machine*, sehingga CVM dapat membuat algoritma *garbage collection* yang lebih singkat dan dapat digunakan secara berulang-ulang.

2.2.1.2. CLDC (Connected, Limited Device Configuration)

CLDC adalah *configuration* yang digunakan pada peralatan *low-end* yang memiliki memori 160-512 Kb, prosesor 16 atau 32 bit, serta konsumsi daya yang kecil. CLDC adalah *configuration* yang dibuat berdasarkan J2SE yang dikurangi beberapa fungsinya. Beberapa fitur J2SE yang dikurangi pada CLDC antara lain adalah :

- *Java Native Interface* (JNI)
- Operasi *floating-point*
- Finalisasi objek
- Paket *java.lang.error*
- *Daemon thread*
- Dan keamanan yang kurang tangguh.

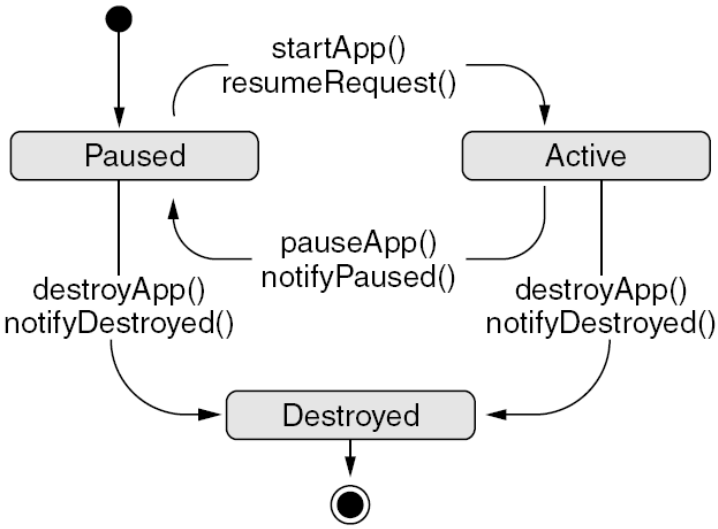
Pada CLDC *virtual machine* yang digunakan adalah KVM (*Kilobyte Virtual Machine*). KVM adalah paket JVM yang didesain untuk perangkat dengan sumber daya yang terbatas. Pada saat ini terdapat CLDC 1.0 dan CLDC 1.1.

2.2.2. Mobile Information Device Profile (MIDP)

Profile adalah bagian dari J2ME yang menyediakan API untuk satu jenis peralatan tertentu, misalnya PDAP (PDA *Profile*) *profile* yang menyediakan API

khusus pada PDA. Salah satu *profile* yang banyak digunakan adalah MIDP yang digunakan pada ponsel, pager, dan Palm OS. Pada saat ini terdapat MIDP 1.0, MIDP 2.0 dan MIDP 2.1, pada MIDP 2.0 terdapat fitur tambahan berupa *Mobile Media API* (MMAPI) yang berfungsi untuk memainkan *file* audio dan video, sedangkan pada MIDP 2.1 terdapat fitur tambahan seperti SVG (*Scalable Visual Graphic*).

Aplikasi yang berjalan pada MIDP disebut Midlet. Midlet terdiri dari minimal sebuah kelas yang merupakan turunan dari kelas abstrak MIDP (*javax.microedition.midlet.MIDLET*). Midlet terdiri dari beberapa metode, yaitu *constructor (), protected void startApp () throws MidletStateChangeException, protected void pauseApp (), protected void destroyApp (boolean unconditional) throws MidletStateChangeException*. Semua metode ini menentukan alur hidup dari midlet yang dibuat. Alur hidup midlet seperti yang ditunjukan pada gambar 2.10, dimulai ketika midlet dijalankan. Awal alur hidup midlet dimulai dengan proses inisiasi midlet ke dalam kondisi *pause* dengan dijalankannya metode *pauseApp (),* lalu midlet dijalankan dengan metode *startApp* (), metode ini diimplememtasikan sebagai *protected* agar midlet yang lain tidak dapat memanggil metode ini. Untuk keluar dari midlet digunakan metode *destroyApp (),* metode ini akan memanggil *notifyDestroyed ()* yang akan memberitahu *platform* utama untuk menterminasi midlet yang sedang dijalankan dan melepaskan semua sumber daya yang digunakan midlet ini agar dapat digunakan midlet lain.



Gambar 2.10. Alur hidup MIDLET [7]

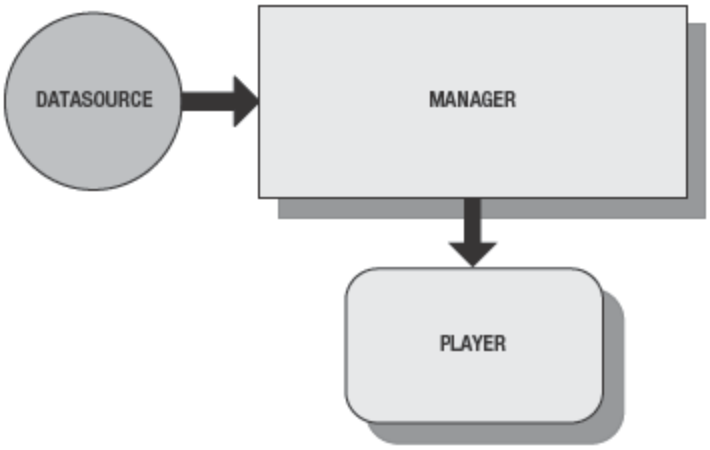
2.2.3. API

Selain *configuration* dan *profile*, J2ME juga dilengkapi beberapa API yang mendukung *platform* ini dengan fungsi-fungsi khusus. Beberapa API yang digunakan dalan skripsi ini adalah MMAPI, *File Connection* API, dan WMA.

2.2.3.1. MMAPI (Mobile Media API)

MMAPI adalah API yang dapat digunakan untuk menambahkan fungsi-fungsi multimedia pada peralatan-peralatan pervasif yang menggunakan J2ME. API ini diperkenalkan melalui JSR 135 (http://www.jcp.org/en/jsr/detail?id=135) oleh *Sun Microsistem*. MMAPI bersifat agnostic terhadap protokol dan format, sehingga API ini dapat digunakan pada sebuah divais tanpa bergantung pada protokol dan format tertentu yang dapat didukung divais tersebut.

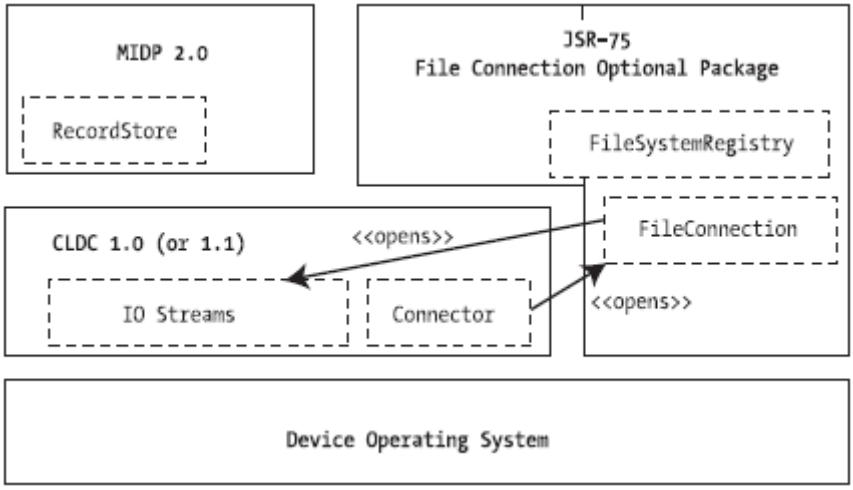
Arsitektur dari MMAPI terdiri dari 3 bagian utama, yaitu *Player*, *Datasource* dan *Manager*. *Player* adalah *interface* yang berkaitan dengan fungsi memainkan dan mengolah data-data multimedia. Sedangkan *DataSource* adalah kelas abstrak yang mengenkapsulasi fungsi pengaksesan data multimedia dan pengaturan protokol yang digunakan. Untuk menjembatani fungsi-fungsi dari *Player* dan *DataSource* diperlukan sebuah *Manager*. Ketiga bagian ini didefinisikan masing-masing pada paket javax.microedition.media, dan javax.microedition.media.protocol. Hubungan antara *Player*, *DataSource* dan *Manager* diilustrasikan pada gambar 2.11.



Gambar 2.11. Hubungan antara *Player*, *DataSource* dan *Manager* [7]

2.2.3.2. File Connection API

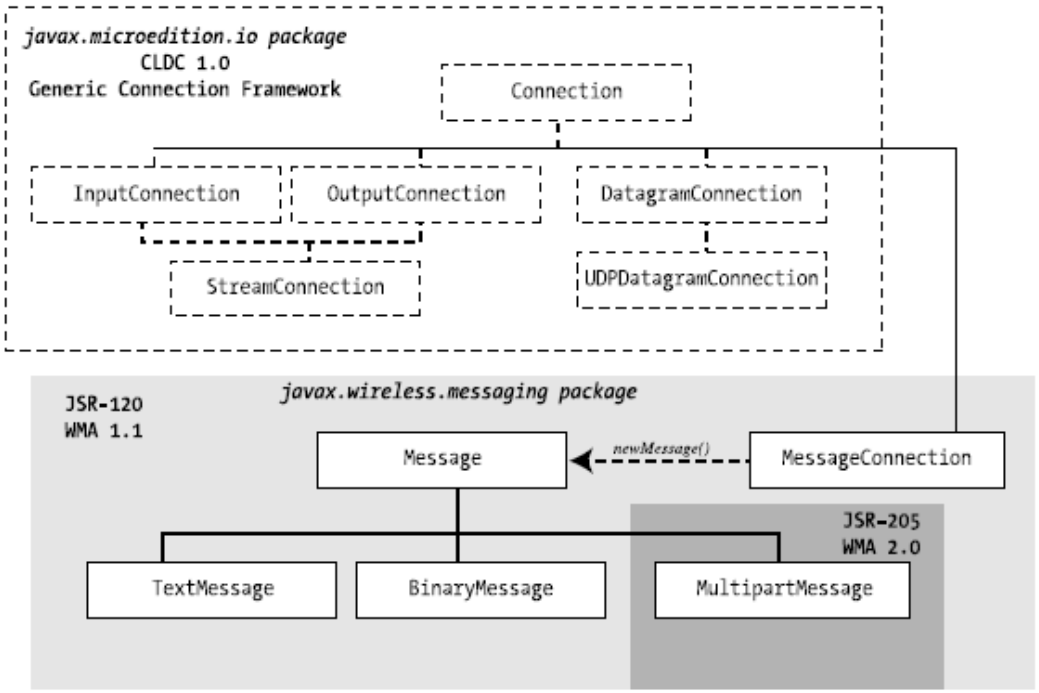
Pada saat ini divais pervasif khususnya ponsel sudah didukung dengan *file* sistem dengan memori yang besar, dengan penambahan *SD card*, *memory stick*, bahkan *hardisk*. *File Connection* API adalah bagian dari PDA *Optional Package* yang diperkenalkan melalui JSR 75 (http://www.jcp.org/en/jsr/detail?id=75) yang berfungsi untuk mengakses *file* sistem dari suatu divais pervasif. API yang di terdapat pada paket javax.microedition.io.*file* ini dapat digunakan untuk membaca dan menulis *file*, serta membuat *file* atau direktori. Hubungan antara API ini dengan *Configuration* dan *Profile* dideskripsikan pada gambar 2.12.



Gambar 2.12. Hubungan antara JSR 75 dengan MIDP dan CLDC [8]

2.2.3.3. WMA (Wireless Messaging API)

WMA adalah API yang dapat digunakan untuk menambahkan fungsi utilitas SMS (*Short Message Sevice*), CBS (*Cell Broadcast Service*) dan MMS (*Multimedia Message Sevice*) pada suatu aplikasi MIDP. Terdapat dua JSR yang berhubungan dengan WMA, yaitu JSR 120 yang menyediakan utilitas SMS dan CBS dan JSR 205 yang menambahkan utilitas MMS pada WMA 2.0. Hubungan antara WMA dan CLDC ditunjukan pada gambar 2.13.



Gambar 2.13. Hubungan antara WMA dan CLDC [8]

BAB III
PERANCANGAN *MOBILE SURVEILLANCE SYSTEM*

3.1. ARSITEKTUR *MOBILE SURVEILLANCE SYSTEM*

Mobile surveillance system yang akan dibangun, berbasiskan pada teknologi *Multimedia Over IP*, dimana video hasil *surveillance* akan dipecah menjadi segmen-segmen pada lapisan trans*port* sesuai dengan UDP dan TCP. Lalu segmen-segmen ini dibentuk menjadi paket-paket yang memiliki alamat IP, yang akan dikirimkan pada jaringan internet. Untuk menjaga reliabilitas dari segmen-segmen ini digunakan protokol RTP, RTSP, dan RTCP pada lapisan aplikasi. Setelah paket-paket ini berada pada jaringan internet, *client* dapat mengambil stream data ini menggunakan ponsel berbasiskan teknologi GSM atau UMTS yang mendukung komunikasi data. Arsitektur sistem ini ditunjukan pada gambar 3.1.

Gambar 3.1 Arsitektur *Mobile surveillance system*

3.1.1. Kamera

Kamera yang pada sistem ini dapat berupa kamera IP, kamera analog maupun webcam. Kamera IP adalah perangkat kamera yang telah memiliki IP sendiri. IP yang dipunyai biasanya merupakan IP *private*. Dengan menggunakan kamera IP, video hasil *surveillance* dapat dikirimkan ke *broadcaster* dengan

menggunakan jaringan IP, baik melalui jaringan Ethernet, maupun jaringan nirkabel. Selain itu dengan menggunakan kamera IP, *broadcaster* dan kamera tidak harus terhubung secara langasung. Sebaliknya apabila digunakan kamera analog atau webcam, kamera harus terhubung langsung ke *broadcaster*, sehingga pemasangan kamera dan *broadcaster* harus berdekatan. Pada kamera analog, hubungan antara kamera dan *broadcaster* dilakukan melalui kabel video biasa, sedangkan pada webcam biasanya digunakan kabel USB. Pada sistem ini akan digunakan *web*cam, yang terhubung langsung ke *broadcaster* dan *streaming server*.

3.1.2. Broadcaster

Setelah video *surveillance* ditangkap oleh kamera, video ini akan dikirimkan ke *broadcaster*. Pada *broadcaster* inilah terjadi proses *encoding*. Pada *broadcaster* ini juga terjadi proses *surestream*, yaitu proses *encoding* sebuah *file* dengan *data rate* yang berbeda-beda. Lalu *file-file* hasil *encoding* ini digabungkan menjadi sebuah *file* yang akan dikirimkan ke *streaming server*. Selain itu, pada *broadcaster* juga dilakukan pembentukan *file-file* SDP yang akan digunakan pada pengiriman data mealui RTSP. Bersama-sama dengan *file-file* SDP, *file* hasil *encoding* dikirimkan ke *streaming server*. Pengiriman *file* ini dapat dilakukan melalui jaringan IP, apabila letak *broadcaster* dan *streaming server* berjauhan. Akan tetapi, hal ini tidak diperlukan apabila *broadcaster* diinstal langsung pada *streaming server*. *Broadcaster* dapat berupa *hardware* maupun *software*. Pada sistem ini akan digunakan *broadcaster* yang berupa *software* yang langsung diinstal padai *streaming server*. *Software broadcaster* yang digunakan adalah Helix *Mobile Producer* 11.1.1. Pada *software* ini dapat diatur tipe *encode*r yang akan digunakan, dan *data rate* untuk mendukung proses *surestream*.

3.1.3. *Streaming Server*

File-file hasil *surestream* dan *file-file* SDP dikirimkan ke *streaming server*. Sebagai sebuah *streaming server* dapat digunakan kompter biasa maupun komputer dengan spesifikasi khusus (*heavy duty*). Dalam melaksanakan fungsinya untuk mengirimkan *live* video *streaming*, *server* ini menggunakan beberapa *port*

pada *transport layer*. *Port-port* yang digunakan ditunjukan pada tabel 3.1. Selain itu hal yang harus diperhatikan adalah, letak dari *server* ini pada jaringan. *Server* ini sebaiknya diletakkan pada DMZ (De-Militarized Zone), sehingga jaringan utama tetap aman, dan dapat menggunakan koneksi yang optimal pada proses *streaming*. Pada sistem ini, digunakan *software* Helix *Server* sebagai *streaming server*.

Tabel 3.1. Port-*port* yang digunakan pada *Mobile surveillance system*

Aktifitas	**Port**	***Transport***	**Fungsi**
Menerima permintaan	554	TCP	Kanal data dan kontrol pada permintaan media melalui RTSP.
Menerima permintaan	80	TCP	Pengaturan transmisi pada HTTP
Menerima permintaan	1755	TCP / UDP	Kanal kontrol pada permintaan media melalui MMS
Menerima permintaan	9090	TCP	Memonitor trafik dan proses *streaming*
Menerima permintaan	4040	TCP	Kanal kontrol dalam penerimaan *file* dari *broadcaster*
Menerima permintaan & Mengirim data	2030	TCP / UDP	Memonitor permintaan pada upstream dan downstream
Menerima permintaan	50001-50050	TCP / UDP	Menerima paket-paket media dari *broadcaster*
Menerima permintaan & Mengirim data	30001-30020	UDP	Kanal data pada RTP
Menerima permintaan	34445-34459	UDP	Menerima jawaban RTP dari *client*, sebelum dilakukan pengiriman ulang

3.1.4. Ponsel

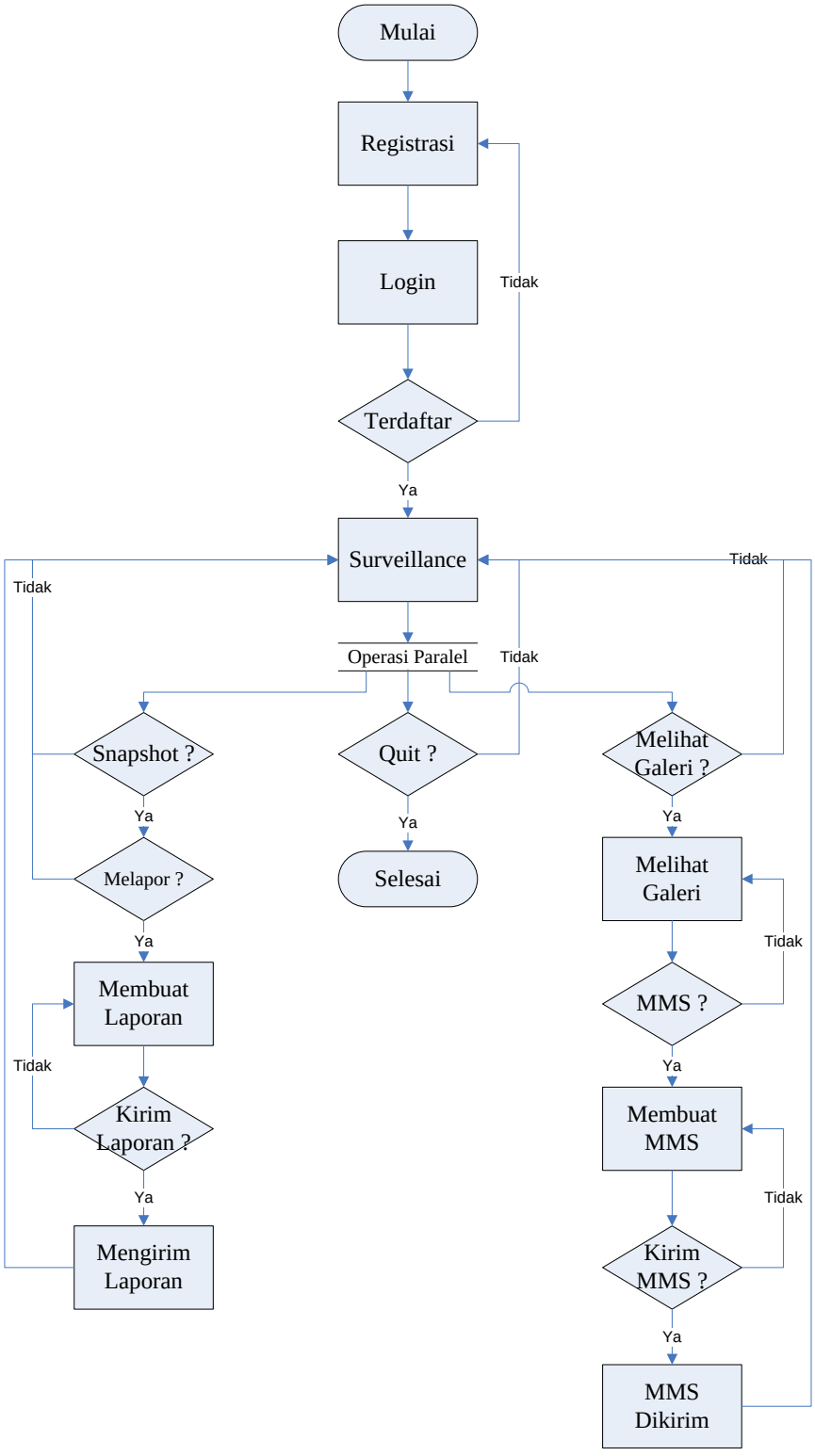
Pada sistem ini *user* dapat menggunakan ponsel yang menggunakan teknologi GSM atau UMTS mendukung komunikasi data dengan kapasitas yang besar dan laju yang cepat. Ponsel yang digunakan harus dapat memainkan video dalam format tertentu, yang sesuai dengan format video hasil keluaran *broadcaster*. Format yang digunakan antara lain .rm dan .3gpp. Agar dapat menjadi bagian dari sistem ini, ponsel yang digunakan harus memiliki *software* yang akan dirancang menggunakan bahasa pemprograman java. Perancangan *software* ini akan dibahas pada sub bab 3.2.

3.2. DIAGRAM ALIR PERANGKAT LUNAK

Diagram alir pada gambar 3.2 menunjukan proses presentasi *live* video *surveillance* pada suatu ponsel. Proses ini adalah salah satu bagian penting dari sistem yang akan dibangun, karana proses inilah yang menentukan bagaimana *user* melihat keadaan beberapa ruangan secara real-*time* melalui ponsel. Penjelasan diagram alir ini adalah sebagai berikut :

1. Awalnya, *user* diminta untuk melakukan registrasi. Lalu *user* diminta untuk melakukan og in, proses ini akan terus dilakukan sampai sistem mengenali *user*. Proses ini adalah salah satu bagian dari fungsi keamanan yang diterapkan pada sistem, dimana hanya *user* yang memiliki otoritas tertentu yang dapat melihat keadaan suatu ruangan melalui sistem ini.
2. Setelah sistem mengenali *user*, maka link akan menjadi aktif secara otomatis. Sehingga *user* dapat melihat keadaan ruangan secara langsung melalui ponselnya.
3. Pada saat *user* melakukan pengintaian, *user* dapat melakukan *snapshot* untuk menangkap gambar dari video yang sedang di*streaming*. Setelah melakukan *snapshot*, *user* dapat langsung melaporkan kejadian yang terjadi. Selain itu, *user* juga dapat melihat semua gambar yang berhasil ditangkap. Gambar-gambar ini juga dapat dikirimkan sebagai MMS.
4. *User* dapat terus mengamati keadaan suatu ruangan sampai ia keluar, yang menandakan selesainya presentasi live video *streaming*.

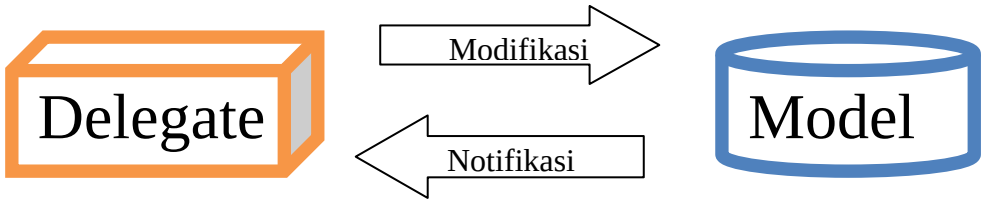
Keempat langkah ini dilakukan dengan dalam sebuah aplikasi midlet yang dibangun pada platform J2ME, sehingga dapat menghasilkan presentasi yang interaktif.



Gambar 3.2. Diagram alir presentasi dari *GySurv*

3.3. UML PERANGKAT LUNAK

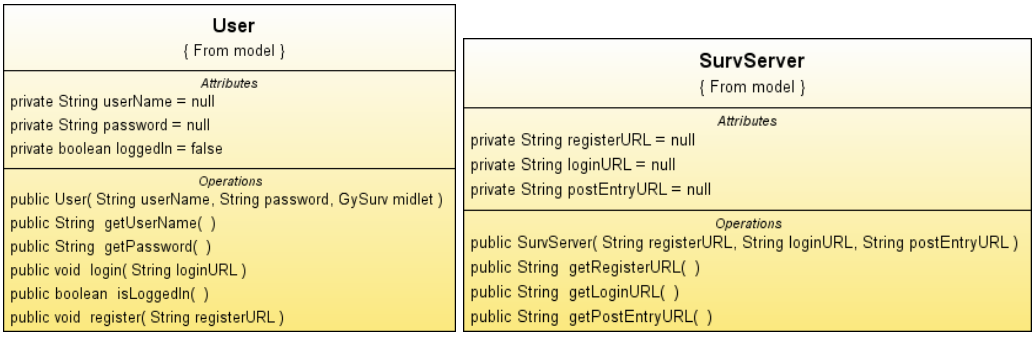
GySurv ini dibangun berdasarkan arsitektur *Delegate-Model* (DM) yang merupakan salah satu varian dari MVC (*Model-View-Control*), dimana *model* berisi data-data aplikasi dari perangkat lunak yang akan dibangun sedangkan *delegate* adalah gabungan dari *view* dan *control* yang merupakan representasi grafis dari *model* dan *interface* untuk memodifikasi *model*. Arsitektur ini ditunjukan pada gambar 3.3. *GySurv* dibangun menggunakan perangkat lunak Netbeans IDE 6.0 dan Sun Java Wireless Toolkit 2.5.2 for CLDC sebagai *compiler* dan *emulator*.



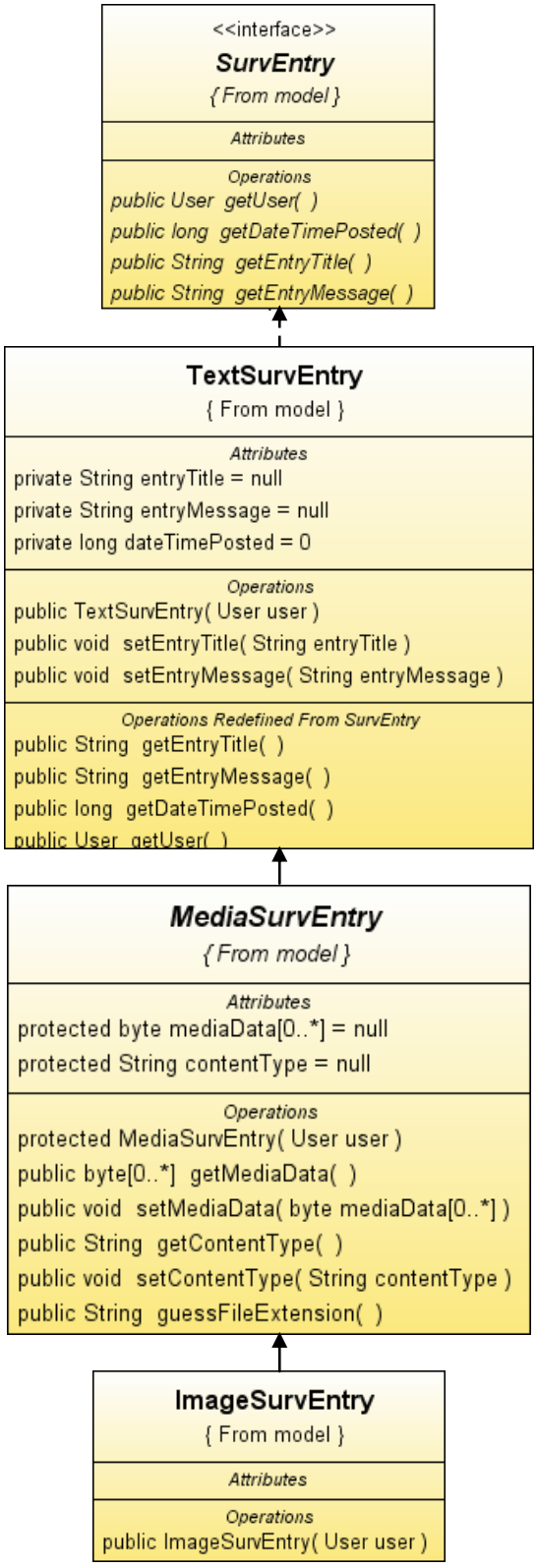
Gambar 3.3 Arsitektur *Delegate-Model*

3.3.1. *Model*

Model dari *GySurv* terdiri dari beberapa kelas yang ditunjukan oleh diagram UML pada gambar 3.4. Kelas *user* berkaitan dengan semua informasi mengenai *user*, sedangkan kelas Surv*Server* berisi URL dari *server* yang digunakan untuk menerima laporan dari *user* ketika terjadi suatu masalah.



Gambar 3.4 *Model* dari *GySurv*

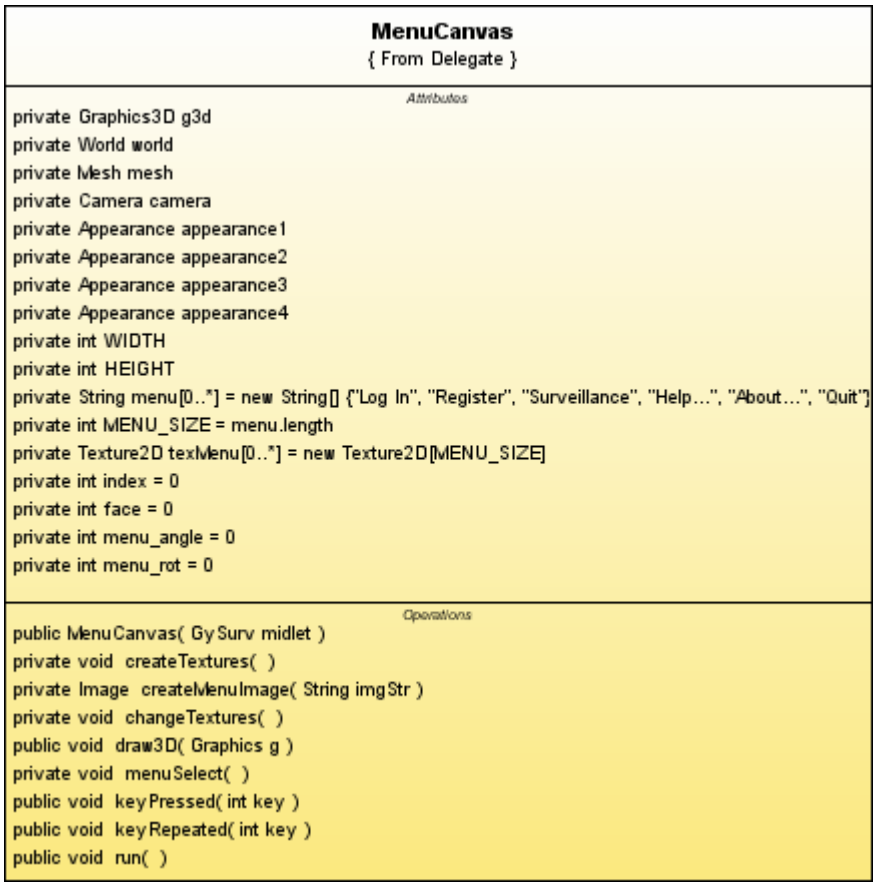


Gambar 3.4 *Model* dari *GySurv* (sambungan dari halaman 27)

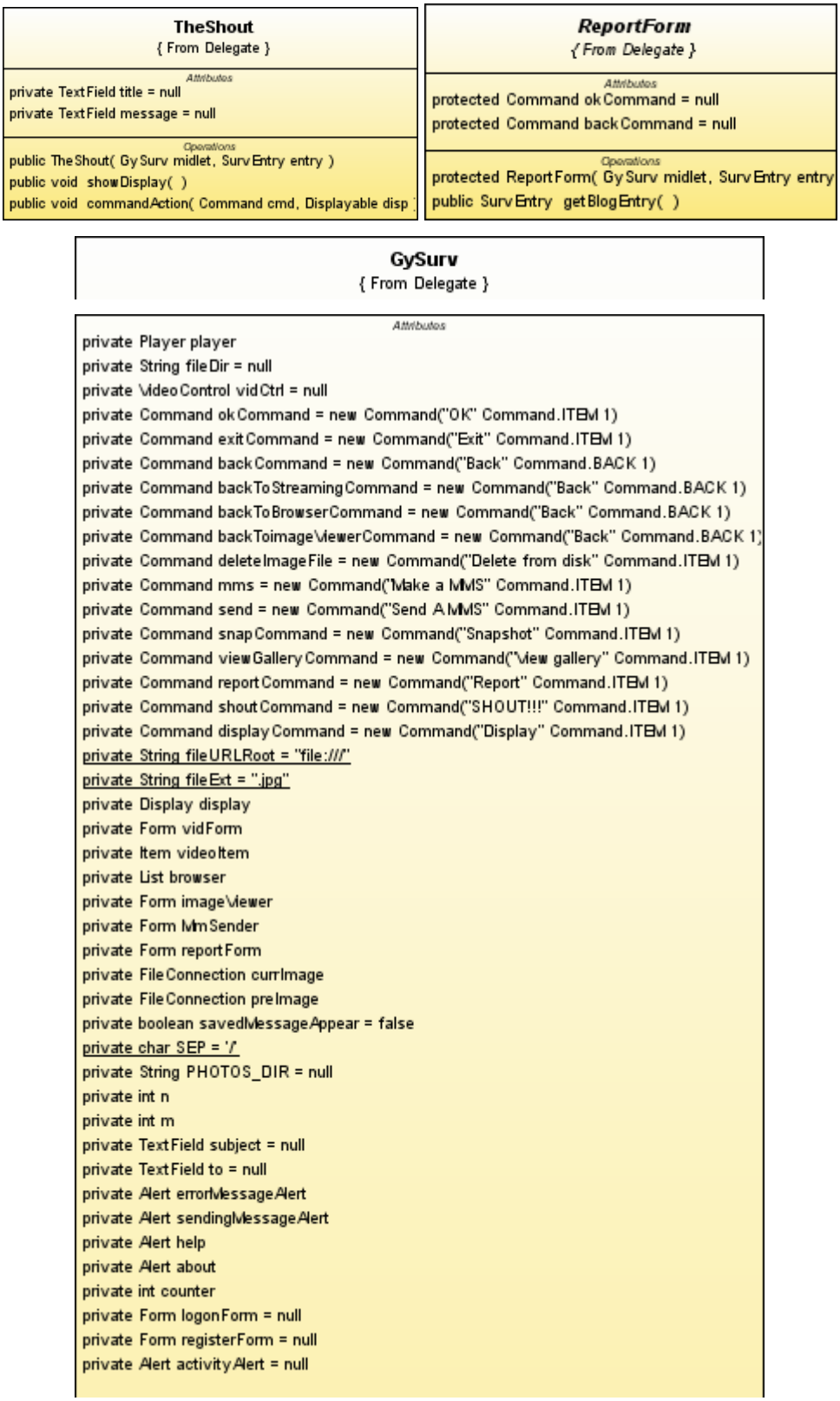
SurvEntry adalah sebuah *interface* yang mendefinisikan semua karakteristik dari entri yang akan dibuat oleh Midlet ini. *Interface* ini terdiri dari empat metode yang harus diimplementasikan oleh keturunannya. Satu-satunya keturunan dari *interface* ini adalah kelas TextSurvEntry yang merepresentasikan entri berbentuk teks. MediaSurvEntry adalah kelas abstrak keturunan dari TextSurvEntry yang menambahkan multimedia sebagi bagian dari entri yang dibuat oleh *GySurv*. Bentuk konkret dari multimedia ini adalah gambar yang diimplemtasikan pada kelas ImageSurvEntry.

3.3.2. Delegate

Seperti yang ditunjukan pada gambar 3.5, delegate terdiri dari kelas Menu Canvas, MC, Alert1, dan StartHScroller yang berungsi untuk menampilkan menu utama dalam tiga dimensi. Keempat kelas ini penulis ambil dari tutorial pada http://developer.sonyericsson.com/getDocument.do?docId=77447.



Gambar 3.5 *Delegate* dari *GySurv*



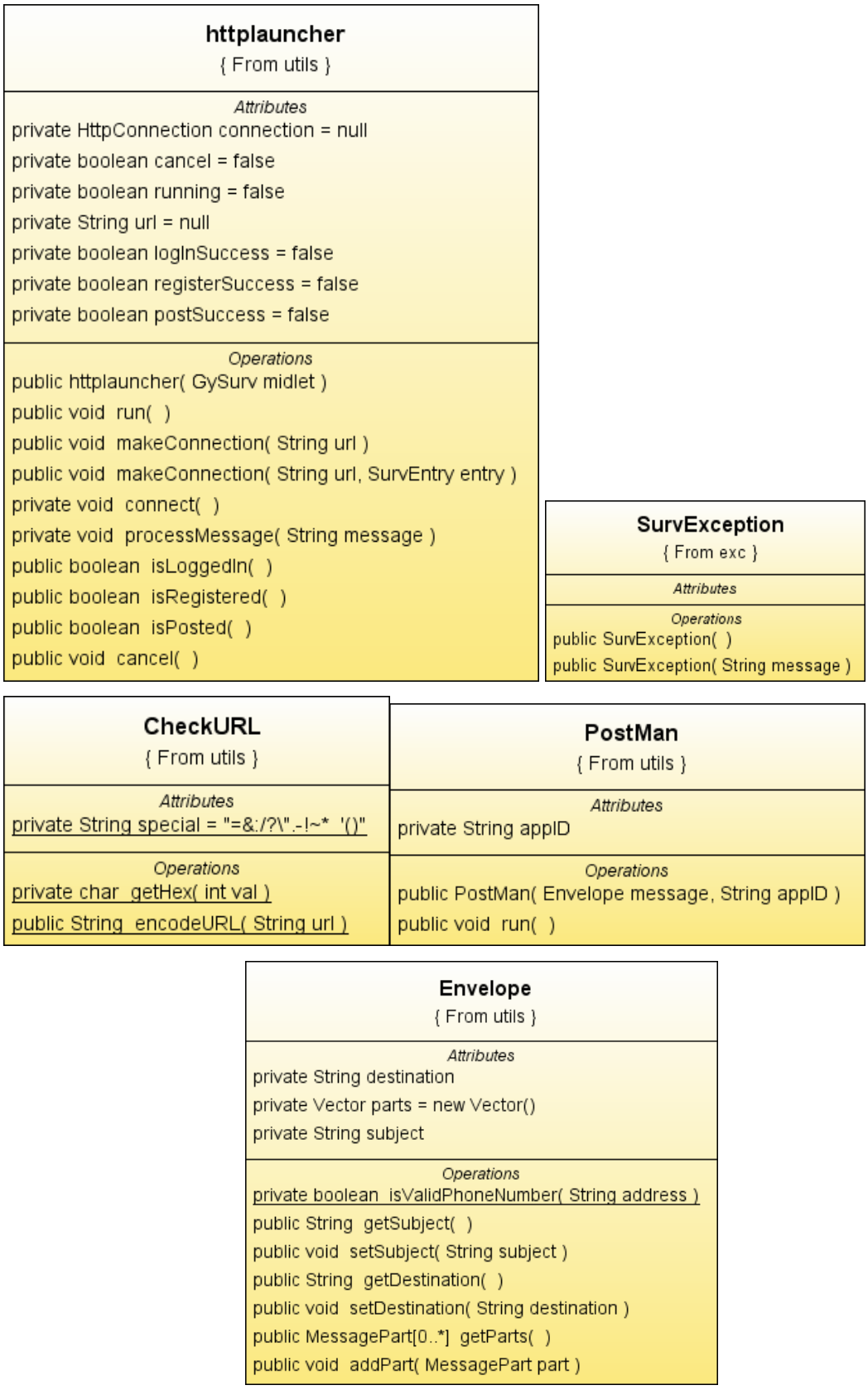
Gambar 3.5 *Delegate* dari *GySurv* (sambungan dari halaman 29)

Gambar 3.5 *Delegate* dari *GySurv* (sambungan dari halaman 30)

Selain itu, terdapat tiga kelas lain yang penulis buat, yaitu kelas *GySurv*, Re*port*Form, dan TheShout yang merupakan inti dari midlet ini. Pada kelas-kelas inilah terdapat fungsi manampilkan video *streaming*, melihat galeri, serta mengirimkan laporan dan MMS.

3.3.3. *Utility*

Pada suatu Java Pattern, selain bagian-bagian utama (MVC atau DM), aplikasi yang dibangun biasanya dilengkapi oleh suatu paket yang berfungsi untuk mendukung bagian-bagian utama tersebut. Paket tersebut adalah *Utility*. Pada *GySurv*, kelas yang berfungsi sebagai *utility* adalah Envelope dan PostMan yang berfungsi untuk mengirimkan MMS, serta kelas httpLauncher dan CheckURL yang berfungsi membentuk koneksi dengan *server*. Selain itu juga terdapat kelas SurvException untuk menunjukan kesalahan yang terjadi pada saat mendebug aplikasi.UML dari *utility* ini ditunjukan pada gambar 3.6.



Gambar 3.6 *Utility* dari *GySurv*

BAB IV
ANALISA DAN UJICOBA

Ujicoba *mobile surveillance* system terdiri dari dua jenis ujicoba. Ujicoba yang pertama adalah ujicoba *surveillance* secara real time, sedangkan ujicoba fitur-fitur tambahan pada *GySurv* dilakukan secara *offline* (*localhost*). Ujicoba *mobile surveillance* system secara real time dilakukan dengan menjalankan *GySurv* pada ponsel yang memanfaatkan jaringan GSM dan UMTS.

Adapun parameter pengujian yang akan dianalisa pada ujicoba ini adalah *delay* serta faktor-faktor yang memperngaruhinya, seperti *data rate* dan *frame rate*. Selain itu akan dilihat juga pengaruh penggunaan jaringan GSM dan UMTS terhadap *delay* yang terjadi. Pengambilan data *delay* dilakukan dengan melihat selisih waktu antara video *streaming* pada ponsel terhadap gerakan yang dilakukan di depan kamera. Pengambilan data ini dilakukan dengan memvariasikan *data rate* dan *frame rate* dari video yang akan di*streaming*.

4.1. SKENARIO UJICOBA

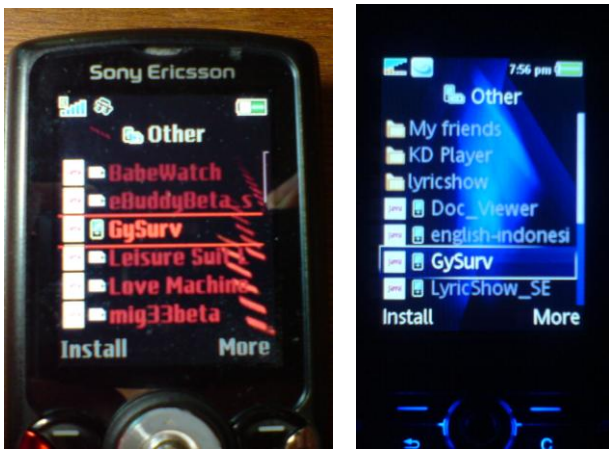
Mobile Surveillance System ini dibangun dari dua subsistem utama, yang dibangun pada sisi *server* dan sisi *client* (ponsel). Pada sisi *server* digunakan *server* dengan sistem operaasi windows *server* 2003 yang didukung dengan perangkat lunak Helix *Server* 11 sebagai *streaming server* dan Helix *Mobile Producer* 11.1.1 sebagai *broadcaster*. Sedangkan pada sisi *client* digunakan ponsel Sonyericsson K810i untuk ujicoba pada jaringan UMTS dan ponsel Sonyericsson W810 untuk ujicoba pada jaringan GSM. Selain itu, pada *Mobile Surveillance* Sistem ini digunakan kamera A4 Tech PK-835. Ujicoba ini memanfaatkan layanan jaringan GSM dan UMTS yang disediakan oleh Telkomsel melalui kartu prabayar Simpati Pede. Alasan penggunaan kartu ini adalah karena layanan 3G yang disediakan operator lain tidak bisa digunakan untuk melakukan *streaming* video.

Kamera dan *server* diletakkan di ruang *server* kantor Pengembangan dan Pelayanan Sistem Informasi Universitas Indonesia (PPSI UI) yang terletak di Fakultas Ilmu Komputer UI (Fasilkom UI). Video yang diambil pada sistem ini adalah keadaan ruang *server* PPSI UI pada tanggal 6 februari 2008 jam 11.00-15.00.

4.1.1. Langkah-langkah instalasi *GySurv*

Sebelum memulai uji coba, *GySurv* harus diinstal pada ponsel yang akan digunakan. Langkah-langkah instalasi yang harus dilakukan adalah sebagai berikut :

1. Awalnya *GySurv*.jar yang telah dibuat menggunakan Netbeans IDE 6.0 disimpan ke dalam ponsel. Pada langkah ini *user* dapat menyimpan *file GySurv*,jar, baik ke memori ponsel maupun ke memori eksternal. Pada ujicoba ini, *file* JAR disimpan ke memori.
2. *Install file* JAR yang telah disimpan ini pada ponsel, instalasi ini juga dapat dilakukan baik pada memori ponsel maupun memori eksternal. Pada ujicoba ini, *file* JAR diinstall pada memori ponsel, di dalam folder other seperti yang ditunjukan pada gambar 4.1.
3. Sebelum menjalankan *GySurv*, jaringan yang akan digunakan pada ponsel harus diatur terlebih dahulu, pengaturan dapat dilakukan pada menu '*setting networks'*. Pada ujicoba ini, jaringan yang digunakan ponsel diatur pada '*GSM only'* untuk ujicoba jaringan GSM dan '*GSM and 3G'* untuk ujicoba pada jaringan UMTS.

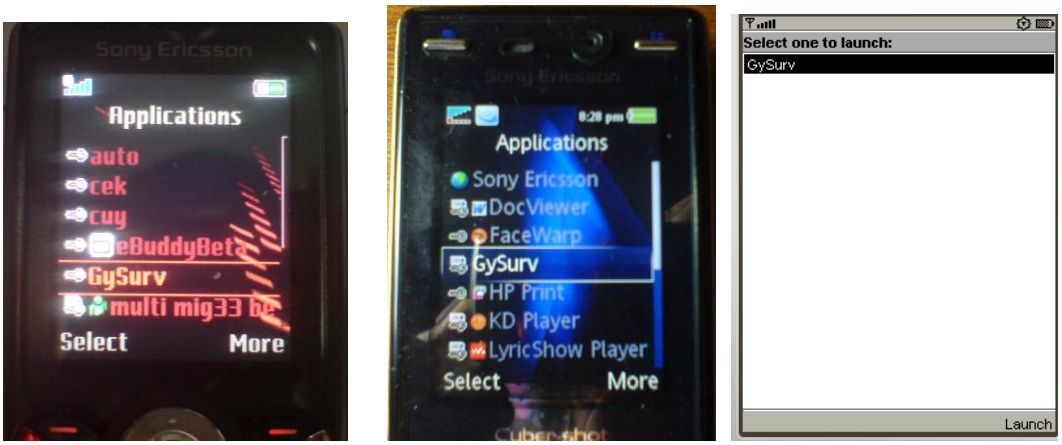


Gambar 4.1 Lokasi penyimpanan *GySurv*.jar pada ponsel

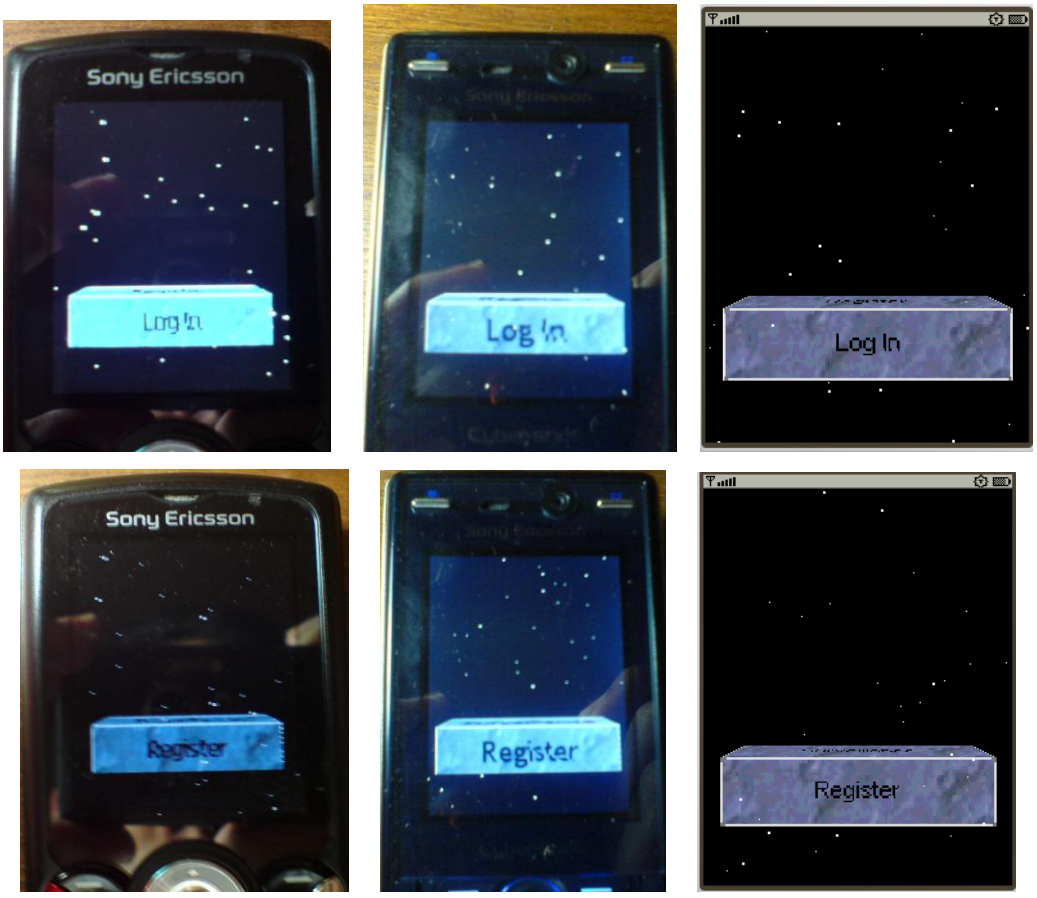
4.1.2. Langkah-langkah menjalankan *GySurv*

Setelah *GySurv* terinstal pada ponsel, maka proses *Surveillance* dapat langsung dilakukan, sesuai dengan langkah-langkah berikut :

1. Jalankan *GySurv* yang telah terinstal pada menu '*application*', seperti yang ditunjukan pada gambar 4.2.
2. Pada saat memulai *GySurv* maka pada layar ponsel akan muncul menu utama seperti yang ditunjukan pada gambar 4.3, pada menu ini *user* dapat melakukan registrasi, *login*, memulai *surveillance*, serta melihat '*help*' dan '*about*'.
3. Sebelum memulai proses *Surveillance user* harus *login* terlebih dahulu pada menu '*log in'*, dan bagi *user* yang belum teregistrasi dapat melakukan proses registrasi terlebih dahulu pada menu '*Register*'. Data-data yang dimasukkan pada proses *login* dan registrasi ini akan dikirimkan ke *server* melalui fasilitas GPRS pada ponsel. Keberhasilan dan kegagalan proses *login* dan registrasi ini ditandai dengan respon dari *server* yang berupa alert. Bila kedua proses ini berhasil, maka alert yang muncul pada layar adalah '*Logged in'* dan *'Registered'*. Apabila *user* melakukan registrasi dengan *user*name yang sudah terdaftar pada *server* maka alert yang muncul pada layar adalah '*Username exists'*, dan apabila *user* melkukan kesalahan pada saat memasukkaa *user*name atau password saat *login*, alert yang akan muncul adalah '*Username not found. Please register'* atau '*Invalid password*'. Proses *Login* dan registrasi ditunjukan pada gambar 4.4.
4. Setelah *user* dikenali oleh *server*, *user* dapat langsung memulai proses *surveillance* dengan memilih menu '*surveillance*' pada menu utama. *User* dapat terus melakukan *surveillance* sampai ia memilih menu '*Quit*' untuk keluar dari aplikasi *GySurv*. Proses ini ditunjukan pada gambar 4.5, berturut-turut pada ponsel sonyericsson W810i, sonyericsson K810i, dan emulator Sun Java Wireless toolkit 2.5.2. Pada saat menggunakan emulator proses ini dilakukan secara offline (localhost) dan gambar diambil dari data asli.



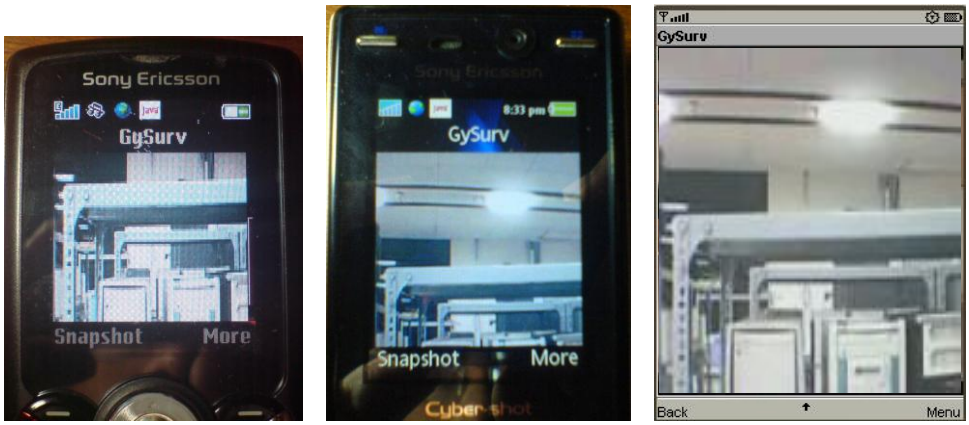
Gambar 4.2 *GySurv* yang telah terinstal



Gambar 4.3 Menu utama pada *GySurv*

Gambar 4.3 Menu utama pada *GySurv* (sambungan dari halaman 37)

Gambar 4.4 Proses *Login* dan registrasi

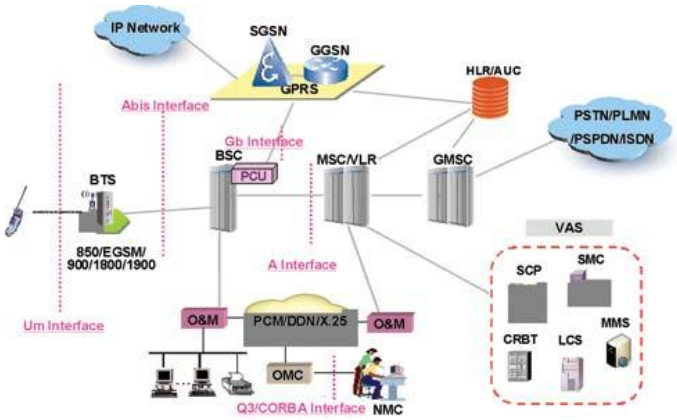


Gambar 4.5 Proses *Surveillance*

4.2. UJICOBA PADA JARINGAN GSM

4.2.1. Topologi pengujian

Pengujian dilakukan dengan menggunakan memanfaatkan layanan GPRS pada topologi jaringan GSM, sehingga ponsel dapat berhubungan dengan jaringan IP. Topologi yang digunakan pada ujicoba ini ditunjukan pada gambar 4.6.



Gambar 4.6. Topologi jaringan GSM [11]

4.2.2. Variasi *data rate*

Variasi *data rate* yang digunakan adalah 24 kbps, 60 kbps, dan 110 kbps. *Data rate* 24 kbps biasanya digunakan untuk mengirimkan data video pada jaringan GSM dan *data rate* 60 kbps digunakan untuk mengirimkan data video pada jaringan EDGE, sedangkan *data rate* 110 kbps digunakan untuk mengirimkan data video pada jaringan UMTS. Keterangan ini ditunjukan oleh *Helix Mobile Producer* pada saat pemilihan *data rate*, seperti yang ditunjukan pada gambar 4.7.

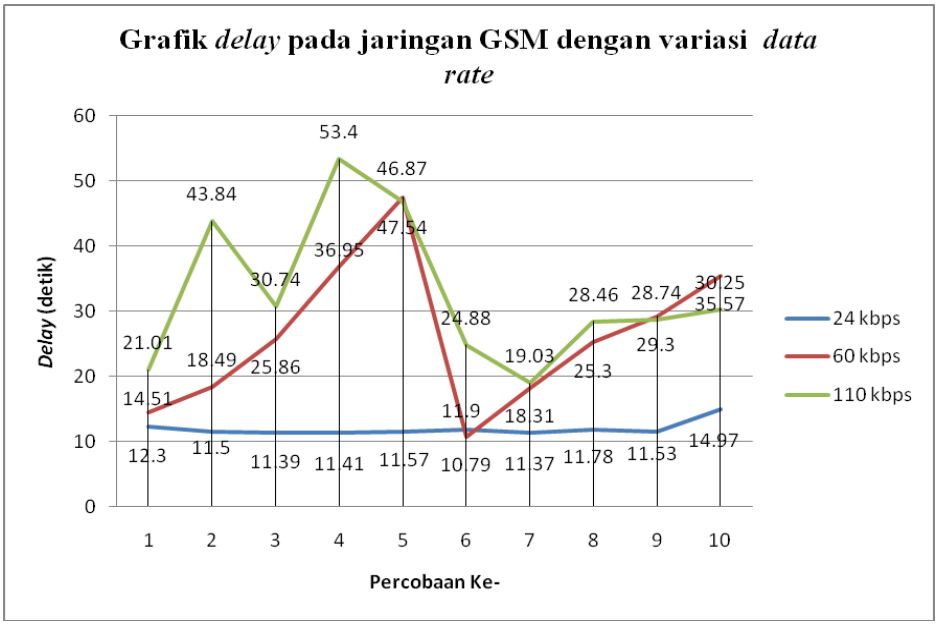
Audience Template	Bit Rate
24k GPRS Voice	24 kbps
50k EDGE Voice	50 kbps
60k EDGE Music	60 kbps
100k UMTS Music	100 kbps
110k UMTS Stereo	110 kbps
225k DSL or Cable	225 kbps

Gambar 4.7 Pemilihan *data rate* pada *Helix Mobile Producer 11.11*

Hasil ujicoba pada jaringan GSM dengan memvariasikan *data rate* ditunjukan pada tabel 4.1 dan grafik perbandingan *delay* dengan 3 variasi *data rate* ditunjukan pada gambar 4.8. Ujicoba ini dilakukan pada *frame rate* 10 fps. Berdasarkan tabel 4.1 dan grafik pada gambar 4.8, dapat dilihat bahwa semakin besar *data rate* yang digunakan maka *delay* yang dihasilkan akan relative meningkat. Selain itu, kualitas video yang dihasilkan saat melakukan ujicoba menunjukan bahwa semakin besar *data rate* yang digunakan akan dihasilkan kualitas video yang lebih baik.

Tabel 4.1. *Delay* pada jaringan GSM dengan variasi *data rate*

Percobaan ke	***Delay* (detik)**		
	24 kbps	**60 kbps**	**110 kbps**
1	12.3	14.51	21.01
2	11.5	18.49	43.84
3	11.39	25.86	30.74
4	11.41	36.95	53.4
5	11.57	47.54	46.87
6	11.9	10.79	24.88
7	11.37	18.31	19.03
8	11.78	25.3	28.46
9	11.53	29.3	28.74
10	14.97	35.57	30.25
Rata-rata (Mean)	11.972	26.262	32.722
Deviasi Rata- rata	0.6652	8.8624	9.1888
Nilai Maksimum	14.97	47.54	53.4
Nilai Minimum	11.37	10.79	19.03



Gambar 4.8 Hasil ujicoba pada jaringan GSM dengan variasi *data rate*

Grafik pada gambar 4.8 menunjukan bahwa *delay* yang dihasilkan pada variasi *data rate* 60 kbps dan 110 kbps cenderung fluktuatif dan tidak stabil. Hal sebaliknya terjadi pada video dengan *data rate* 24 kbps, hal ini dikarena *data rate* yang cocok pada jaringan GSM adalah pada 24 kbps. Selain itu, kualitas video

yang dihasilkan pada *data rate* 60 dan 110 kbps sangat buruk, hal ini terlihat dari kualitas video yang tidak stabil dan tidak mulus.

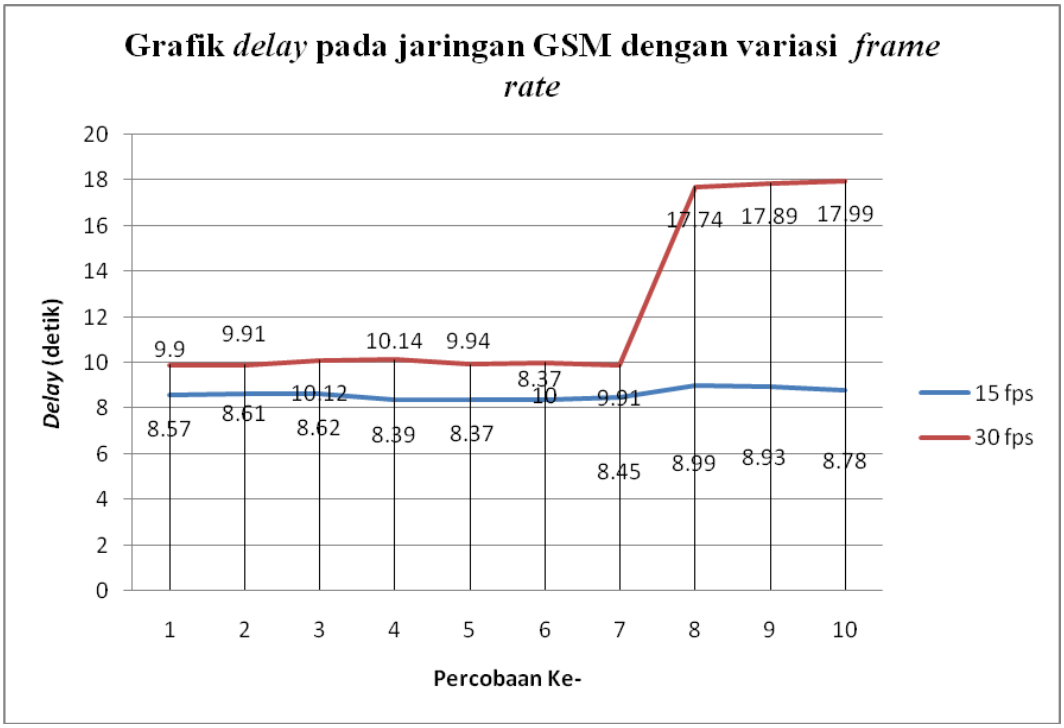
4.2.3. Variasi *frame rate*

Variasi *frame rate* yang digunakan pad ujicoba ini adalah 15 fps dan 30 fps. Pemilihan *frame rate* sebesar 30 fps disesuaikan standar yang digunakan oleh *National Television Systems Committee* (NTSC) yang digunakan di amerika, sedangkan alasan pemilihan *frame rate* sebesar 15 fps adalah untuk melihat pengaruh nilai *frame rate* ini terhadap *delay* yang terjadi. Nilai *frame rate* sebesar 30 fps pada ujicoba ini menghasilkan video yang lebih mulus dibandingkan dengan *frame rate* 15 fps. Hasil ujicoba pada jaringan GSM dengan memvariasikan *frame rate* ditunjukan pada tabel 4.2 dan grafik perbandingan *delay* dengan 2 variasi *frame rate* ditunjukan pada gambar 4.9. Ujicoba ini dilakukan pada *data rate* 24 kbps yang sesuai untuk jaringan GSM.

Tabel 4.2. *Delay* pada jaringan GSM dengan variasi *frame rate*

Percobaan ke	***Delay* (detik)**	
	15 fps	**30 fps**
1	8.57	9.9
2	8.61	9.91
3	8.62	10.12
4	8.39	10.14
5	8.37	9.94
6	8.37	10
7	8.45	9.91
8	8.99	17.74
9	8.93	17.89
10	8.78	17.99
Rata-rata (Mean)	8.608	12.354
Deviasi Rata- rata	0.178	3.3116
Nilai Maksimum	8.99	17.99
Nilai Minimum	8.37	9.9

Pada ujicoba ini *frame rate* harus diatur sebelum kamera merekam keadaan ruangan, pengaturan ini dilakukan pada *broadcaster*, dalam hal ini Helix *Mobile* Producer.

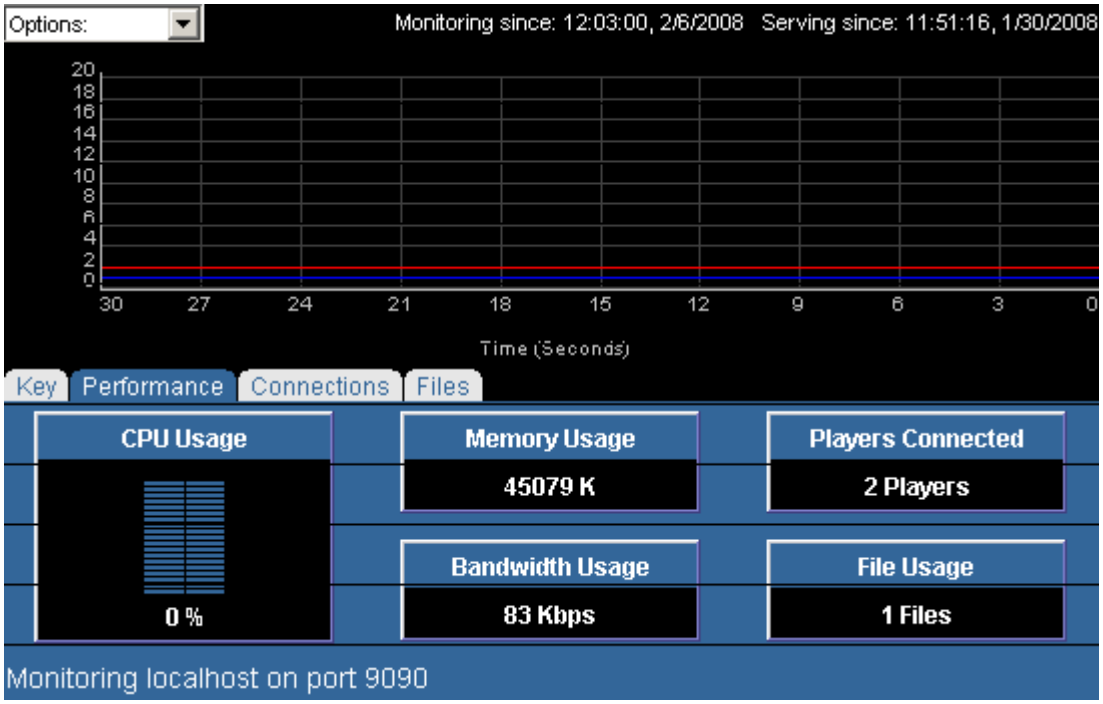


Gambar 4.9 Hasil ujicoba pada jaringan GSM dengan variasi *frame rate*

Dengan melihat hasil ujicoba pada jaringan GSM ini, dapat diambil kesimpulan bahwa nilai *data rate* dan *frame rate* yang optimal untuk jaringan ini adalah 24 kbps dan 15 fps. Dengan nilai ini dihasilkan *delay* rata-rata 10.326 detik, dan video dengan kualitas tidak terlalu rendah. Delay yang terjadi ini jauh lebih kecil daripada delay yang terjadi pada system multimedia Over IP yang berbasiskan mikrokontroller yaitu sebesar 18.443982 detik [12].

4.2.4. *Bandwidth*

Untuk melihat *bandwidth* yang digunakan pada video *streaming*, digunakan fasilitas *monitoring* pada helix *server*. Selain *bandwidth*, fasilitas ini juga menunjukan besarnya memori pada *server* yang digunakan untuk mengirimkan video. Pada gambar 4.10 dapat dilihat bahwa *bandwidth* yang digunakan adalah sebesar 83 kbps. Besarnya *bandwidth* yang digunakan tidak banyak terpengaruh terhadap variasi *data rate* dan *frame rate* yang digunakan.

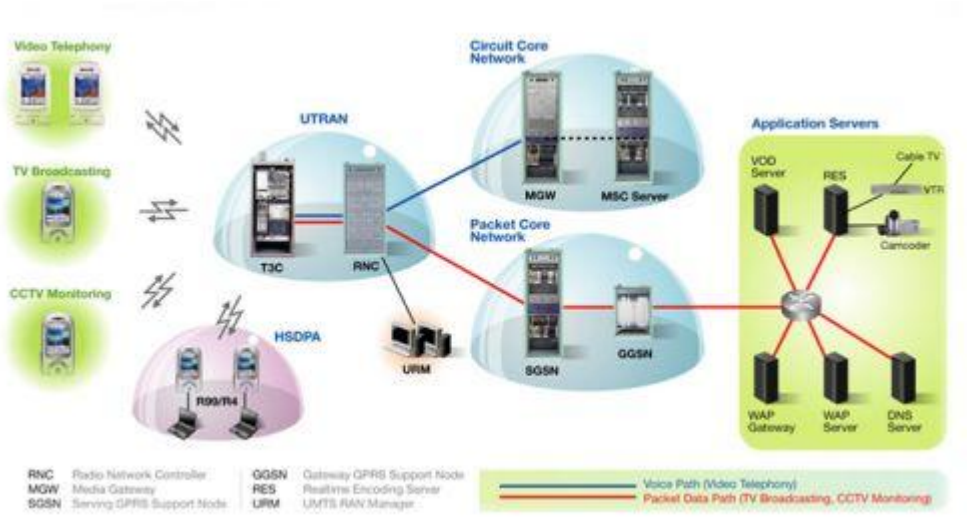


Gambar 4.10. *Bandwidth* yang digunakan pada jaringan GSM

4.3. UJICOBA PADA JARINGAN UMTS

4.3.1. Topologi pengujian

Topologi yang digunakan pada ujicoba ini ditunjukan pada gambar 4.11.



Gambar 4.11. Topologi jaringan UMTS [9]

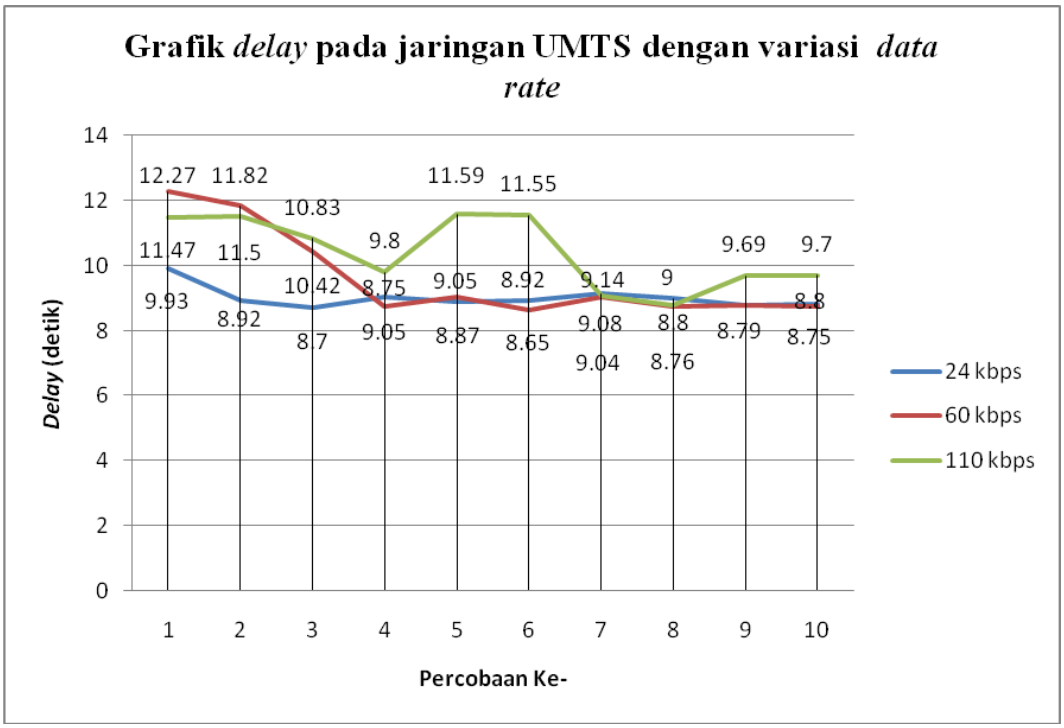
4.3.2. Variasi *data rate*

Ujicoba pada jaringan UMTS dilakukan dengan variasi *data rate* yang sama dengan ujicoba jaringan GSM. Hal ini bertujuan untuk melihat perbandingan *delay* antara kedua jaringan ini. Tabel 4.3 dan gambar 4.12 menunjukan data hasil ujicoba yang dilakukan pada jaringan UMTS.

Tabel 4.3. *Delay* pada jaringan UMTS dengan variasi *data rate*

Percobaan ke	**Bit *Rate***		
	24 kbps	**60 kbps**	**110 kbps**
1	9.93	12.27	11.47
2	8.92	11.82	11.5
3	8.7	10.42	10.83
4	9.05	8.75	9.8
5	8.87	9.05	11.59
6	8.92	8.65	11.55
7	9.14	9.04	9.08
8	9	8.76	8.8
9	8.79	8.79	9.69
10	8.8	8.75	9.7
Rata-rata (Mean)	9.012	9.63	10.401
Deviasi Rata- rata	0.2168	1.124	0.987
Nilai Maksimum	9.93	12.27	11.59
Nilai Minimum	8.7	8.65	8.8

Dari hasil ujicoba pada tabel 4.3 dapat dilihat bahwa *delay* yang terjadi ketika menggunakan jaringan UMTS relative lebih kecil dibandingkan *delay* yang terjadi pada jaringan GSM. Hal ini dikarenakan jaringan UMTS didukung dengan bit *rate* dan *bandwidth* yang lebih tinggi dibandingkan jaringan GSM. Selain itu pada ujicoba juga ditunjukan bahwa video dengan *data rate* 110 kbps menghasilkan video yang lebih mulus dibandingkan dengan video dengan *data rate* 60 kbps dan 24 kbps, begitu pula video dengan *data rate* 60 kbps dihasilkan video yang lebih baik dibandingkan dengan video dengan *data rate* 24 kbps.



Gambar 4.12 Hasil ujicoba pada jaringan UMTS dengan variasi *data rate*

Pada ujicoba ini pengaturan *data rate* dilakukan pada *broadcaster* yitu Helix *Mobile* Producer, pengaturan ini dilakukan sebelum video diambil dan dikirim ke *server*.

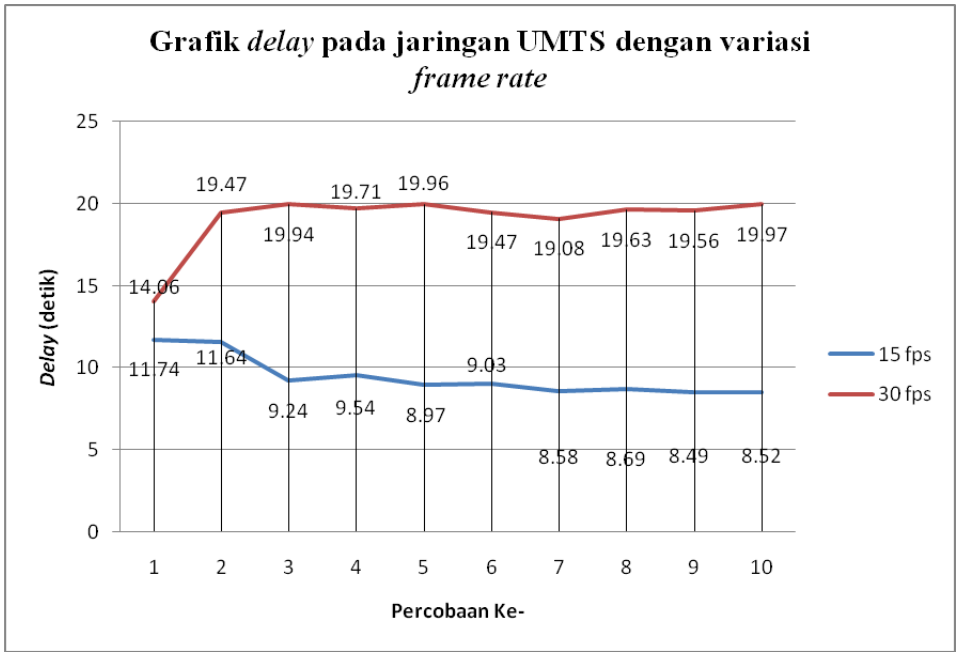
4.3.3. Variasi *frame rate*

Pada ujicoba ini *frame rate* yang digunakan adalah 15 fps dan 30 fps, sama seperti ujicoba pada jaringan GSM. Akan tetapi, ujicoba ini dilakukan pada *data rate* 110 kbps yang sesuai untuk jaringan UMTS. Hasil ujicoba ditunjukan pada tabel 4.4 dan gambar 4.12.

Berdasarkan tabel 4.4 dan grafik pada gambar 4.13, dapat dilihat bahwa pemilihan *frame rate* 15 fps menghasilkan *delay* yang lebih rendah dari pada *frame rate* 30 fps. Perbandingan kualitas antara video dengan 15 fps dan video dengan 30 fps menunjukan hasil yang relative sama pada saat ujicoba dilakukan. Hal ini menunjukan bahwa pemilihan video dengan *frame rate* 15 fps pada jaringan UMTS lebih efisien dibandingankan dengan video dengan *frame rate* 30 fps.

Tabel 4.4. *Delay* pada jaringan UMTS dengan variasi *frame rate*

Percobaan ke	***Frame rate* pada 60 Kbps**	
	15 fps	**30 fps**
1	11.74	14.06
2	11.64	19.47
3	9.24	19.94
4	9.54	19.71
5	8.97	19.96
6	9.03	19.47
7	8.58	19.08
8	8.69	19.63
9	8.49	19.56
10	8.52	19.97
Rata-rata (Mean)	9.444	19.085
Deviasi Rata- rata	0.9176	1.006
Nilai Maksimum	11.74	19.97
Nilai Minimum	8.49	14.06

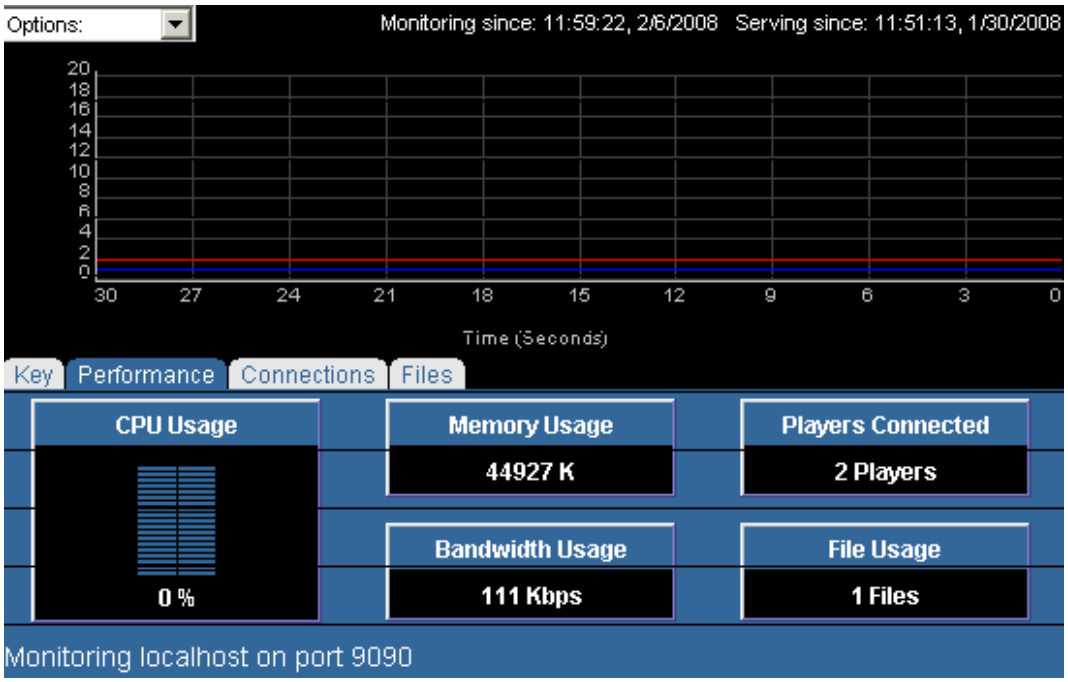


Gambar 4.13 Hasil ujicoba pada jaringan UMTS dengan variasi *frame rate*

Dengan melihat hasil ujicoba pada jaringan UMTS ini, dapat diambil kesimpulan bahwa nilai *data rate* dan *frame rate* yang optimal untuk jaringan ini adalah 110 kbps dan 15 fps. Dengan nilai ini dihasilkan *delay* rata-rata 9.9225 detik, dan video dengan kualitas yang jauh lebih baik daripada video yang dihasilkan saat menggunakan jaringan GSM. Delay yang terjadi ini jauh lebih kecil daripada delay yang terjadi pada system multimedia Over IP yang berbasiskan mikrokontroller yaitu sebesar 18.443982 detik [12].

4.3.4. *Bandwidth*

Pada pengiriman video melalui jaringan UMTS, *bandwidth* yang digunakan adalah sebesar 111 kbps seperti yang ditunjukan gambar 4.14.



Gambar 4.14. *Bandwidth* yang digunakan pada jaringan UMTS

4.4. UJICOBA *START UP TIME*

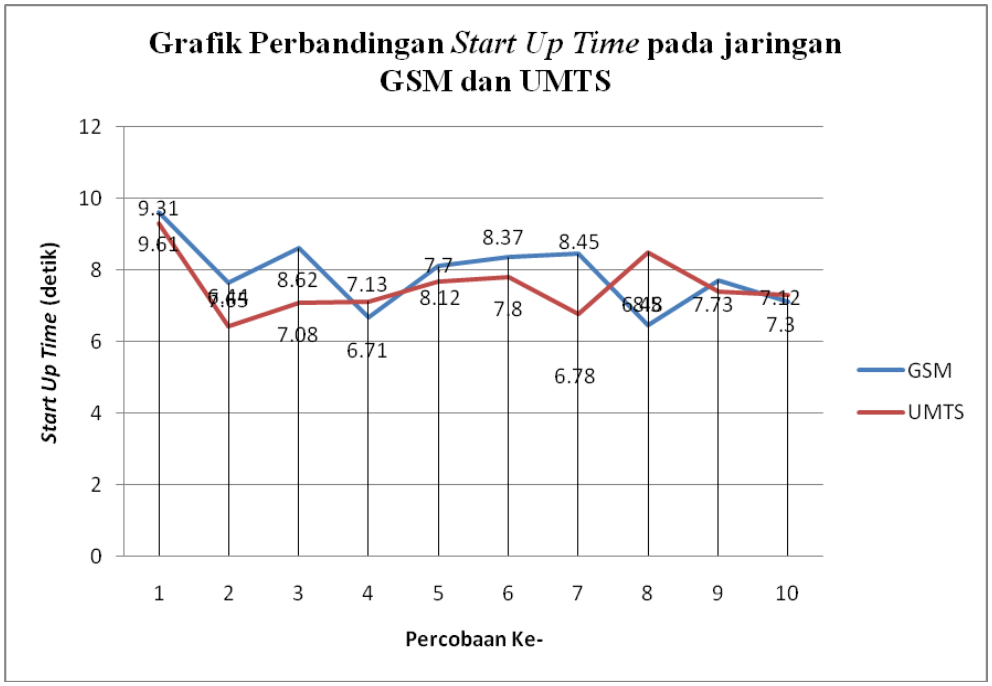
Pada ujicoba ini diambil data berupa *start up time* yang merupakan waktu yang diperlukan oleh *GySurv* untuk terhubung ke streaming server. Tabel 4.5 dan gambar 4.15 menunjukan hasil ujicoba ini.

Pada tabel 4.5 dan grafik 4.15 dapat dilihat bahwa *start up time* yang dibutuhkan oleh *GySurv* apabila dijalankan pada jaringan GSM dan UMTS tidak

jauh berbeda. Hal ini dikarenakan arsitektur yang diguankan oleh jaringan GSM dan UMTS untuk mengakses jaringan IP tidak jauh berbeda.

Tabel 4.5. *Start Up Time* pada jaringan GSM dan UMTS

Percobaan ke	***Start Up Time* (detik)**	
	GSM	**UMTS**
1	9.61	9.31
2	7.65	6.44
3	8.62	7.08
4	6.71	7.13
5	8.12	7.7
6	8.37	7.8
7	8.45	6.78
8	6.48	8.5
9	7.73	7.4
10	7.12	7.3
Rata-rata (Mean)	7.886	7.544
Deviasi Rata- rata	0.748	0.6268
Nilai Maksimum	9.61	9.31
Nilai Minimum	6.48	6.44



Gambar 4.15. Perbandingan *Start Up Time* pada jaringan GSM dan UMTS

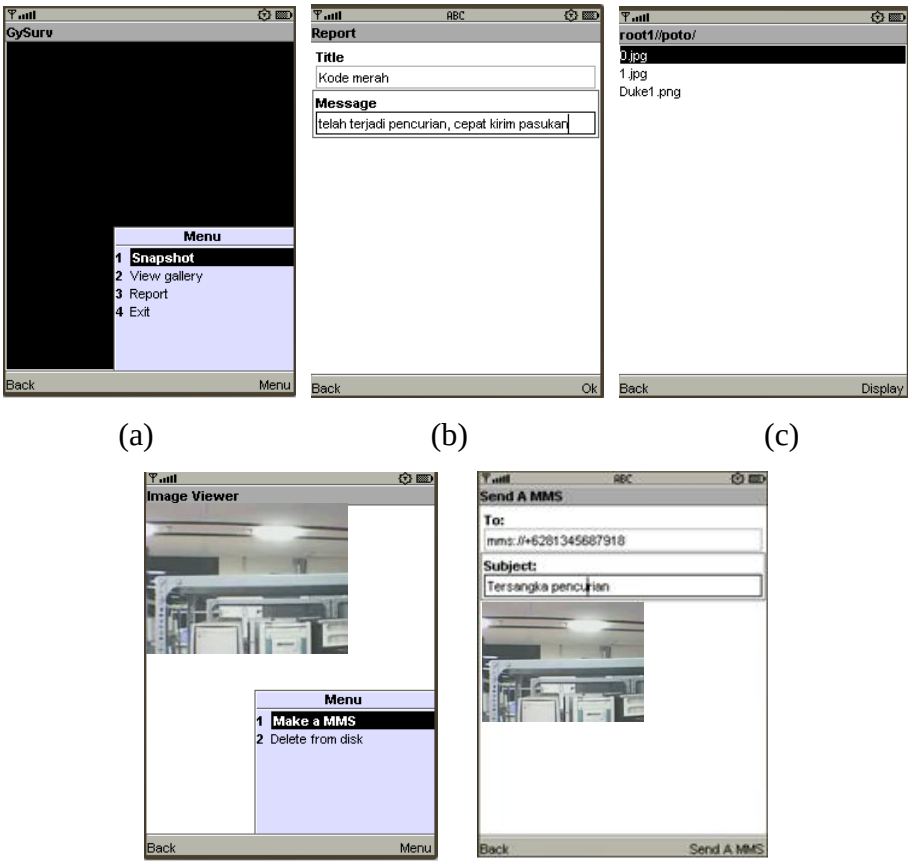
4.5. UJICOBA FITUR-FITUR TAMBAHAN

Mobile Surveillance System yang dibangun pada *client* memilikibeberapa fitur tambahan, seperti menyimpan foto hasil *snapshot* video yang sedang di*streaming*, melihat galeri dari foto-foto hasil *snapshot* ini, serta mengirimkan foto-foto ini sebagai MMS. Selain itu foto-foto ini dapat dikirimkan ke sebuah *server*, misalnya *server* kepolisian untuk meminta pertolongan. Ujicoba ini dilakukan secara *offline* (*localhost*) dikarenakan tidak adanya sertifikat yang diperlukan untuk ujicoba secara real time. Sertifikat ini dibutuhkan untuk menngakses *file system* pada ponsel yang dilakukan dengan menggunakan *File Connection* API. Sertifikat yang dapat digunakan untuk mengaktifkan fitur-fitur ini antara lain sertifikat Verisign *class* 3, Thawte *premium server*, serta Geotrust CA for UTI.

Adapun langkah-langkah yang harus dilakukan untuk melakukan ujicoba ini sama seperti langkah-langkah menjalankan *GySurv* yang telah disebutkan sebelumnya dengan tambahan beberapa langkah sebagai berikut :

1. Pada saat layar menampilkan video *surveillance*, tekan command menu, pada command ini terdapat beberapa pilihan aksi yang bisa dilakukan, seperti yang ditunjukan pada gambar 4.16 (a). Untuk melakukan *snapshot* pilih '*Snapshot*', untuk meluhat galeri pilih *'View galery'*, dan untuk mengirimkan laporan ke *server* kepolisian pilih '*Report*'.
2. Command '*Snapshot*' adalah command yang sangat penting disini. Karena sebelum *user* dapat mengirimkan laporan, *user* harus melakukan *snapshot* terlebih dahulu, dan begitu pula untuk melihat galeri, karena foto yang dilihat pada galeri adlah foto hasil *snapshot*.
3. Setelah *snapshot* dilakukan, *user* dapat langsung mengirimkan laporan. Setelah command re*port* dipilih maka akan muncul form yang harus diisi *user* sebelum mengirimkan re*port*. Form ini terdiri dari judul laporan serta isi laporan, seperti yang ditunjukan pada gambar 4.16 (b). Setelah semuanya terisi *user* dapat mengirimkan laporan ke *server* pengaduan.
4. Selain itu *user* juga dapat melihat foto-foto hasil *snapshot* yang telah dilakukan sebelumnya, seperti yang ditunjukan pada gambar 4.16 (c) dan

(d). Pada saat melihat galeri, *user* dapat mengirimkan foto yang sedang ia lihat sebagai MMS dengan menekan command '*Make MMS'*. Setelah command ini dipilih maka akan muncul form yang harus diisi *user*, seperti yang ditunjukan pada gambar 4.16 (e). Form ini terdiri dari nomor tujuan MMS. Serta pesan yang terkait dengan MMS ini.



(a) (b) (c)

(d) (e)

Gambar 4.16 Ujicoba fitur-fitur tambahan

4.6. BIAYA

Pada ujicoba ini, pengambilan data yang dilakukan tidak hanya berupa data teknis seperti *delay* dan *bandwidth*, melainkan juga meliputi data ekonomis, berupa biaya yang digunakan dalam ujicoba ini. Biaya (*Cost*) adalah data yang sangat penting yang akan digunakan sebagai salah satu pertimbangan dalam pengimplementasian *Mobile Surveillance* System ini sebagai sebuah produk. Dengan mengetahui biaya yang harus dikeluarkan, maka nilai jual dari *Mobile Surveillance* System ini dapat diestimasi.

Ujicoba dilakukan masing-masing selama sekitar satu jam untuk jaringan GSM dan jaringan UMTS. Pengambilan data tidak dilakukan secara terus menerus, dalam hal ini proses *surveillance* tidak dilakukan secara kontinu selama satu jam, akan tetapi dilakukan bebarapa kali. Hal ini terutama terjadi pada ujicoba jaringan GSM, karena sering terjadi putusnya hubungan dengan jaringan IP, saat ujicoba dilakukan dengan *data rate* 60 kbps dan 110 kbps. Pada ujicoba *Mobile Surveillance* System ini biaya yang dikeluarkan adalah sesuai dengan besarnya data yang dikirimkan dengan fasilitas GPRS melalui jaringan Telkomsel, yaitu Rp 12/kb baik untuk jaringan UMTS maupun untuk jaringan GSM.

Pada ujicoba untuk jaringan UMTS, biaya yang dikeluarkan untuk pengambilan data untuk durasi sekitar satu jam adalah Rp. 160.000,00 atau sebesar 13.333 Kb. Sedangkan ketika digunakan jaringan GSM untuk durasi waktu yang sama dibutuhkan biaya sebesar Rp. 300.000,00. Pembengkakan biaya yang harus dikeluarkan ketika menggunakan jaringan GSM ini terjadi, terutama saat sistem dipaksakan untuk mengambil video *streaming* dengan *data rate* yang tinggi, yang lebih cocok digunakan untuk jaringan UMTS.

Berdasarkan hasil ujicoba ini, maka secara ekonomis *Mobile Surveillance* sistem ini lebih cock diterapkan dengan menggunakan jaringan UMTS, karena untuk dihasilkan video *streaming* dengan gambar yang baik dan tidak terputus-putus dibutuhkan biaya yang lebih murah dibandingkan dengan menggunakan jaringan GSM.

4.7. PEMANFAATAN SISTEM

Sistem ini dibangun untuk mendukung sistem kemanan yang ada, dengan menambahkan fitur mobilitas. Selain itu sistem ini juga mendukung pengiriman data baik berupa teks maupun gambar yang diambil dari video yang sedang di*streaming* melalui MMS. Dengan fitur-fitur ini, *mobile surveillance system* dapat dimanfaatkan dalam berbagai layanan *Quad Play* lainnya.

4.7.1. Media *distant-learning*

Dengan meletakkan kamera pada ruang-ruang kelas, sistem ini dapat digunakan untuk mengirimkan materi yang diajarkan dosen didepan kelas. Sehingga siswa dapat mengikuti perkuliahan tanpa harus hadir di ruang kelas.

4.7.2. Pemantau lalu-lintas

Sistem ini dapat digunakan untuk memantau lalu-lintas dengan menempatkan kamera pada lokasi-lokasi yang st*rate*gis. Hasil pantauana ini dapat digunakan sebagai pertimbangan oleh pengguna dalam memilih rute yang akan dilaluinya.

BAB V
KESIMPULAN

1. *Mobile surveillance system* ini dapat berjalan dengan baik dan dapat dimanfaatkan untuk berbagai keperluan.
2. *Delay* rata-rata yang terjadi saat sistem diterapkan pada jaringan GSM adalah 23,652 detik, sedangkan *delay* rata-rata yang terjadi apabila sistem diterapkan pada jaringan UMTS adalah 9,681 detik.
3. Untuk mengurangi *delay* yang terjadi, terdapat beberapa faktor yang dapat diatur, yaitu *data rate* dan *frame rate*.
4. Pada jaringan GSM, nilai *data rate* dan *frame rate* optimal ayng dapat diterapkan adalah 24 kbps dan 15 fps, sedangkan pada jaringan UMTS nilai *data rate* dan *frame rate* yang optimal adalah 110 kbps dan 15 fps.
5. Dalam penerapan sistem pada jaringan UMTS dihasilkan kualitas video yang jauh lebih baik, dengan harga yang lebih murah dibandingkan dengan penerapan sistem pada jaringan GSM.

DAFTAR ACUAN

[1] Tore Terjesen, "Applikasjonsutvikling - mobile tjenester", Thesis, Norwegian university of science and technology, Faculty of electrical engineering and telecommunications, Malaga Spanyol, 2001, hal 18.

[2] Harri Holma and Antti Toskala, *WCDMA for UMTS : Radio Access for Third Generation Mobile Communication, Third Edition* (West Sussex: John Wiley & Sons Ltd, 2004), hal 7.

[3] Jack Y. B. Lee, *Scalable Continuous Media Streaming* Sistems (West Sussex: John Wiley & Sons Ltd, 2005), hal 112, 114, 115.

[4] David Austerberry, *The Technology of Video and Audio Streaming* (Jordan Hill: Focal Press, 2005), hal 219, 210, 214.

[5] Amitabh Kumar, *Mobile TV DVB-H, DMB, 3G* Systems *And Rich Media Applications* (Jordan Hill: Focal Press, 2007), hal 67, 69.

[6] *Helix Server Administration Guide* (Seattle: RealNetworks, Inc, 2007), Chapter 2

[7] Digia. Inc, *Programming for the Series 60 Platform and Symbian OS* (Helsinki: John Wiley & Sons Ltd, 2003), hal 426, 443.

[8] Vikram Goyal, *Pro Java ME MMAPI Mobile Media APIfor Java Micro Edition* (New York: Apress, 2006), hal 4

[9] Sing li and Jonathan Knodsen, *Beginning J2ME: From Novice to Profesional, Third Edition* (New York: Apress, 2005), hal 118 dan 167

[10] Future UMTS Network architecture, http://www.mypdacafe.com/upload_*files*/mpc_articles/233/hsdpa_1.jpg, diakses 11 Februari 2008 jam 12.15

[11] ZTE's GSM Solution in action,http://wwwen.zte.com.cn/main/newspic/20067247303920.jpg, diakses 11 Februari 2008 jam 12.18

[12] Adri Gautama, "*System Multimedia Over IP* berbasis *Embeded Ethernet* Mikrokontroler", Thesis, Pasca Sarjana Teknik Elekro UI, Depok, 2007, hal 77.

DAFTAR PUSTAKA

Demetriades, Gregory C., *Streaming Media: Building and Implementing Complete Streaming* Sistem (Indianapolis: Wiley Publishing, Inc, 2005)

Hofmann, Markus, Leland R. *Beaumont, Content Networking: Architecture, Protocols, and Practice* (San Francisco: Morgan Kaufmann Publishers, 2005)

Parkins, Colin, *RTP: Audio and Video for the Internet* (Boston: Addison Wesley, 2003)

Rayburn, Dan, Michael Hoch, *The Business of Streaming and Digital Media* (Jordan Hill: Focal Press, 2005)

Topic, Michael, *Streaming Media Demystified* (USA: McGraw-Hill Telecom, 2002)