PENGAPLIKASIAN MEDIA *STREAMING* PADA *M-LEARNING* DENGAN MENGGUNAKAN *SELULAR* CDMA

Gatot Santoso[1]

ABSTRACT

Telecommunications technology and internet technology represent a most technology liked in the world. In this technological growth enable new breakthrough in learning by mobile use IT peripheral grasp or referred as mobile learning (m-learning). M-Learning have some excess among others ability learn " every where and any where". Existing Problem the peripheral study of m-learning have limitation and many resource of] platform so that needed device capable to guarantee compatible between is assorted platform.

This scheme aim to as learning media which is not limit room and time.

Keywords : *m-learning, streaming* CDMA

INTISARI

Teknologi telekomunikasi dan teknologi internet merupakan sebuah teknologi yang paling digemari di seluruh dunia. Dalam perkembangan teknologi ini memungkinkan terobosan baru dalam belajar secara *mobile* menggunakan perangkat IT genggam atau disebut *mobile learning* (*m-learning*). *M-Learning* memiliki beberapa kelebihan di antaranya adalah kemampuan belajar “kapan-pun di mana-pun”. Problem yang ada adalah perangkat pembelajaran *m-learning* memiliki keterbatasan sumber daya dan keragaman *platform* sehingga diperlukan rancangan yang mampu menjamin kompatibilitas antara berbagai macam *platform*.

Perancangan ini bertujuan sebagai media pembelajaran yang tak terbataskan ruang dan waktu.

Kata Kunci: m-*learning*, *streaming*, CDMA.

PENDAHULUAN

Dalam melakukan pengaplikasian media streaming pada *M-learning* dengan menggunakan *selular* CDMA, terdapat dua buah teori dasar yaitu teknologi *streaming* dan teknologi CDMA. Streaming berasal dari bahasa Inggris, yang artinya pengaliran atau mengalirkan. Maka dalam dunia internet, *streaming* mengacu kepada teknologi yang mampu mengkompresi atau menyusutkan ukuran *file* audio dan video agar mudah ditransfer melalui jaringan internet.

Pentransferan file audio dan video tersebut dilakukan secara mengalir te-rus-menerus. Dari sudut pandang prosesnya, *streaming* berarti teknologi pengiriman *file* dari *server* ke *client* melalui jaringan *packet-based* semisal internet. *File* tersebut berupa serangkaian paket yang diberi stempel waktu yang disebut stream.

Dari sudut pengguna, *streaming* adalah teknologi yang memungkinkan *file* dapat segera dijalankan tanpa harus menunggu selesai di*download* seluruhnya. Sebelum teknologi *streaming* diperkenalkan luas, *file* tersebut harus di-*download* secara utuh baik *file* audio atau video sebelum dapat mendengar atau menontonnya di komputer. Untuk men-*download file* tersebut hingga selesai tentu saja memerlukan waktu yang cukup lama. Sekedar contoh, jika sebuah *file* video besarnya adalah 10 MB, maka diperlukan waktu sekitar 15 menit jika menggunakan akses internet dengan kecepatan 56 Kbps.

Padahal, menurut beberapa *survey*, batas kesabaran rata-rata pengguna internet untuk menunggu ditayangkannya sesuatu yang diakses hanyalah 8 detik saja. Lebih dari itu, mereka akan meninggalkan situs tersebut. Pada Februari 1999, Lucasfilm meluncurkan potongan film *Star Wars* di situsnya. Besar *file* trailer tersebut adalah 50 Mb, trafik dan antrian pen-*download* saat itu benar-benar sangat tinggi hingga 48 jam ke depan. *File* tersebut belum menggunakan teknologi

[1] Staf jurusan Teknik Elektro, FTI, ISTA Yogyakarta

streaming, sehingga kebanyakan orang yang tidak dapat men-*download* secara sempurna *file* tersebut dan akhirnya sama sekali tidak dapat menikmati trailer tersebut. Kini dengan semakin luasnya penggunaan teknologi *streaming*, maka kendala-kendala tersebut di atas sudah dapat diatasi.

Pada metode *download* konten itu ditaruh pada suatu *server*, misalnya *web server*. Untuk dapat menggunakannya, klien akan men-*download* seluruh *file* dan disimpan pada *hard disk* lokal. Jika aplikasi ini akan dipresentasikan, maka dibutuhkan suatu media *player* yang tepat. Keuntungan dari metode ini karena *file* disimpan di*hard disk*, sehingga kapan saja dibutuhkan *file* ini dapat dimainkan tanpa harus koneksi kejaringan serta kualitas dari konten tidak tergantung dengan kondisi jaringan. Namun kerugian dari metode ini adalah dibutuhkan waktu yang lama untuk men*download* dan membutuhkan tempat penyimpanan pada *hard disk* lokal.

Pada metode *streaming*, klien mempersentasikan konten yang datang dari jaringan secara langsung, tanpa men-*download* seluruh konten terlebih dahulu. Konten *streaming* ini tidak di*download*, tetapi paket konten ini dipresentasikan kemudian dibuang. Keuntungan dari konten *streaming* ini adalah cocok untuk durasi konten yang tidak terbatas waktunya misalkan untuk acara yang sifatnya *live*, seperti internet radio dan TV *on demand*. Kerugian dari metode ini adalah kualitas dari konten tergantung dari kondisi *bandwidth* jaringan. Kondisi jaringan yang buruk dan fluktuasi *bandwidth* akan menghasilkan gangguan yang sangat berarti pada kualitas presentasi.

Menurut Ariwibowo (2003) *streaming* dapat dibagi menjadi dua subkategori, yaitu *on demand stream* dan *web cast stream*.

a. *On demand stream*

Berikut adalah cirri-ciri dari *on demand stream* :

1. *On demand stream* dikontrol oleh klien
2. *On demand stream* diaktifkan oleh permintaan *user* dan dapat dipresentasikan kapan saja sesuai dengan perintah klien
3. *On demand stream* ini dapat dimisalkan seperti video-kaset, kita dapat melakukan *fast-forward*, *rewind*, *pause* dan lainnya.

b. *Web caststream*

Berikut adalah cirri-ciri dari *Web caststream* :

1. *Webcast stream* hanya dapat mengontrol apakah akan terus menerima konten atau tidak.
2. Jika *webcast* adalah suatu peristiwa yang *live*, *user* harus melakukan hubungan pada *server webcast stream* untuk melihat konten sesuai dengan jadwal yang telah ditentukan.
3. *Webcast stream* dapat dimisalkan seperti kita melihat televisi atau mendengarkan radio. Jadi yang mengatur konten presentasinya adalah stasion TV atau radio tersebut.

Teknik *streaming* ini terbagi menjadi dua, yaitu HTTP *Streaming* dan *True Streaming.*

a. Teknik HTTP (*Hyper Text Transfer Protocol*) *streaming*

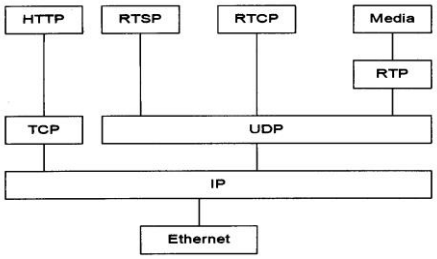
HTTP *streaming* disebut juga *web server streaming* karena menggunakan protokol HTTP untuk mengirimkan *file*nya. Sebelum dikirim ke klien biasanya *file* dikompres terlebih dahulu menjadi tipe *file* media tertentu, misalnya format *real media* (.rm) untuk *realplayer* dan *realone player* atau *advanced streaming format* (.asf) untuk *windows media player*, kemudian dihubungkan menggunakan URL ke *file* tersebut. Teknik *streaming* ini cocok untuk konten multimedia berukuran kecil. Oleh karena HTTP menggunakan TCP untuk mentransfer data yang bersifat *reliable*, maka kemungkinan besar akan terjadi *delay*. Disamping itu, dalam protocol HTTP tidak mendukung interaksi dua arah untuk mengontrol *streaming*, seperti pengaturan pada *bandwidth*, *rewind*, *pause* atau *fast-forward*.

b. Teknik *true streaming*

Pada teknik *true streaming* digunakan protokol UDP (*User Datagram Protocol*) untuk transfer konten multimedia ke klien. Pada UDP tidak

diperiksa apakah data telah diterima atau belum dan tidak mengirim ulang paket data yang rusak atau hilang, karena UDP adalah protokol untuk pengiriman data yang bersifat *unreliable*, tidak seperti pada TCP. Dengan demikian, protokol ini cocok untuk transfer konten multimedia yang terus-menerus, misalnya *live event*. Sedangkan pada tingkat aplikasi dapat digunakan protokol RTSP atau *real-time protocol* (RTP). RTP menambahkan informasi *header* pada paket UDP, seperti misalnya *time-stamp*, *sequence number* dan tipe kompresi agar dapat dilakukan timing sinkronisasi, pengurutan dan *decoding* paket-paket pada sisi tujuan. Disamping RTP juga digunakan protokol RTCP (*Real Time Control Protocol*) untuk mengontrol pendistribusian data.

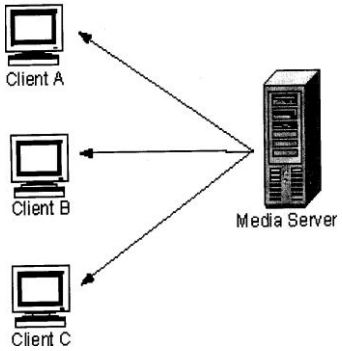
RTSP (*real time streaming protocol*) merupakan protokol aplikasi yang dapat melakukan pengontrolan kualitas layanan (QoS) konten multimedia yang ditransfer ke klien, dengan pengontrolan, seperti *pause*, *stop*, *rewind* dan *fast-forward*. Beberapa protokol tadi dapat digambarkan sebagai berikut:



Gambar 1. Protokol yang digunakan pada teknik streaming

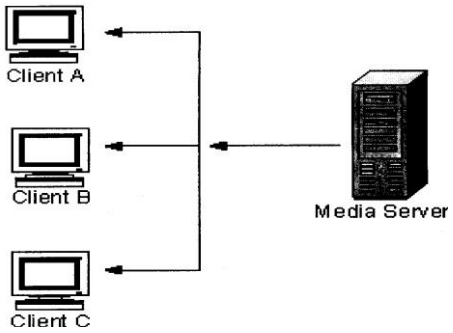
Unicast dan *multicast* adalah metode *transport* pada jaringan yang menggambarkan bagaimana menyampaikan konten ke klien. Kedua metode tersebut digunakan dalam *streaming*.

a. *Unicast*
Transmisi *unicast* merupakan transmisi informasi yang dilakukan dari satu pengirim ke satu penerima. Transmisi ini juga sering dikenal dengan transmisi *point to point.* Setiap penerima akan memperoleh *stream* yang berbeda walaupun menampilkan *file* yang sama. *Unicast* digunakan pada on-*demand streaming* karena setiap klien mempunyai hubungan dua arah dengan *server*. Keuntungan adalah adanya hubungan dua arah dengan *server*, sehingga memungkinkan mengirim informasi *control* dan *feedback* ke *server* yang dapat digunakan untuk *error correction* dan adaptasi terhadap kondisi jaringan. Kerugiannya adalah jika *unicast streaming* melayani klien yang sangat banyak, akan mempengaruhi *bandwidth* yang digunakan. Misalnya, jika *bandwidth* konten adalah 100 Kbps dan ada 1000 klien, jumlah *bandwidth* jaringan yang dibutuhkan satu *server* adalah 100 Mbps.



Gambar 2. Sistem transmisi *unicast*

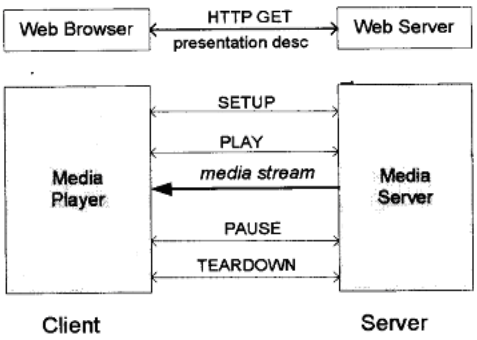
b. *Multicast*
Transmisi *multicast* merupakan transmisi dari satu pengirim ke banyak penerima. Setiap penerima akan menerima stream yang sama. *Multicast* adalah metode transmisi data secara *connectionless*, yang berarti klien menerima aliran data tapi tidak terhubung secara langsung ke *server*. Metode ini menghemat *bandwidth* jaringan karena hanya satu aliran data yang dibangkitka oleh *server*. Sebagai contoh terdapat 3 buah penerima yang meminta transmisi informasi sebesar 100 kb/s, maka total *bandwith* yang dibutuhkan tetap 100 kb/s. Di dalam jaringan terdapat *router* yang dapat melakukan *multicast* paket-paket aliran data ini. *Multicast* digunakan pada *webcast stream*, misalnya internet radio, dan tidak dapat digunakan untuk pengiriman *on-demand stream*.



Gambar 3 Sistem transmisi *multicast*

Protokol RTSP

RTSP (*real time streaming protocol*) adalah protokol pada tingkat aplikasi untuk mengontrol penyampaian data secara *real-time*. Dengan RTSP klien dapat mengontrol jalannya presentasi, misalnya melakukan *play*, *rewind*, *pause, resume, stop* atau *fast-forward* terhadap aliran data. Sumber aliran data dapat meliputi keduanya, *webcast* dan *on-demand*. Protocol RTSP memiliki *sintaks* dan operasi yang mirip dengan protokol HTTP/ Gambar 1. Operasi protokol RTSP ini dapat digambarkan seperti berikut:



Gambar 4. Operasi protokol RTSP

Operasi protokol RTSP berdasarkan alokasi dan penggunaan sumber media (*resource*) di *server* memiliki beberapa *state* :

SETUP : *Server streaming itu* mengalokasikan *resource* dan suatu *session* pada RTSP dimulai

PLAY dan RECORD: Mentransmisikan data media yang dialokasikan via SETUP

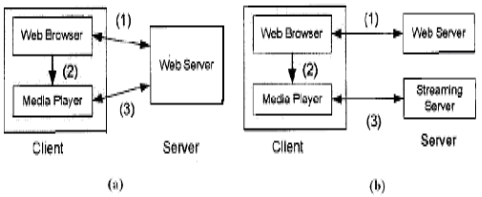
PAUSE : Media *player* yang dalam keadaan berhenti sementara, tetap menggunakan *resource* di *server*

TEARDOWN : Membebaskan *resource* yang berhubungan dengan media player tersebut, dan *session* RTSP dibuang dari *server*

Disamping itu, ada juga protokol MMS (*Microsoft Media Server*) yang merupakan protokol aplikasi yang digunakan untuk mengakses konten yang sifatnya *unicast* dari *windows media server*. Protokol MMS berisi mekanisme *control* untuk menangani *request* dari klien seperti "*play*" atau "*stop*" dan mekanisme pengiriman data yang menjamin bahwa paket data diterima dalam format yang dikenali oleh klien. Protokol TCP digunakan untuk membawa *request* (*signaling*), sedangkan paket data dibawa menggunakan protokol TCP atau UDP.

Pada sistem *streaming* media dan komponen-komponennya bisa digambarkan seperti berikut:

1. Media *source* dapat berupa sumber yang sifatnya *live*, seperti kamera atau *microphone*
2. *Encoder* adalah program yang digunakan untuk mengubah media *source* ke format yang sesuai untuk *streaming*. Biasanya memiliki kompresi yang cukup tinggi untuk mengatasi keterbatasan *bandwidth* jaringan.
3. Media *server* digunakan untuk mendistribusikan *on-demand* atau *webcast* suatu konten ke klien. Juga bertanggung jawab untuk mencatat semua aktivitas *streaming*, yang nantinya digunakan untuk *billing* dan *statistic*. Implementasinya dapat menggunakan *web server* (HTTP *streaming*) atau *streaming server* (*true streaming*). Perbedaan *web server* dengan *streaming server* seperti pada gambar di bawah:



Gambar 5. (a) *Web server* dan (b) *Streaming server*

Web server :

1. *Web browser* melakukan *request* dan menerima *metafile* (suatu *file* yang

mendeskripsikan objek) melalui protokol HTTP.
2. *Browser* menentukan *player* yang cocok sesuai dengan informasi dalam *metafile* dan meneruskan *metafile* ke media *player* tersebut.
3. Media *player* melakukan hubungan TCP dengan *web server*. Data *stream* di-*request* dan dikirim menggunakan protokol HTTP.

Streaming server :

1. *Web server* melakukan *request* dan menerima *metafile* atau *file* yang mendeskripsikan presentasi melalui protokol HTTP.
2. *Browser* menentukan *player* yang cocock sesuai dengan informasi dalam *metafile* ke media tersebut.
3. *Player* melakukan hubungan TCP atau UDP dengan *streaming server*. Data *stream* di-*request* dan dikirim menggunakan protokol RTSP.

Player dibutuhkan untuk menampilkan atau konten multimedia (data *stream*) yang diterima dari media *server*. *File-file* khusus yang disebut *metafile* digunakan untuk mengaktifkan *player* dari halaman WWW. *Metafile* berisi keterangan dari konten multimedia. *Browser* WWW men-*download* dan meneruskan ke *player* yang tepat untuk mempresentasikannya. Fungsi lainnya adalah melakukan dekompresi.

Beberapa masalah yang terdapat pada *streaming* diantaranya adalah :

1. *Bandwidth*
   Bandwidth itu juga sangat berpengaruh terhadap kualitas presentasi suatu data *stream*. Disamping kondisi jaringan juga mempengaruhi *bandwidth*, hal yang perlu diperhatikan adalah ukuran data stream harus sesuai dengan kapasitas pada *bandwidth* jaringan tersebut. Untuk mengatasinya, digunakan kompresi data dan penggunaan *buffer*.
2. Sinkronisasi dan *delay*
   Sinkronisasi bertujuan agar suatu media yang berbeda dapat sampai ke *user* dan dapat dipresentasikan pada *user* seperti aslinya, agar sesuai dengan *timeline* presentasi tersebut dengan menggunakan *delay* yang seminimal mungkin. Adanya kerugian sinkronisasi dan *delay* dapat disebabkan oleh kondisi jaringan yang buruk, sehingga mengakibatkan *timeline* presentasi menjadi kacau.
3. *Interoperability*
   Idealnya adalah presentasi yang kita buat harus dapat dimainkan oleh semua jenis klien, CPU yang berbeda, sistem operasi yang berbeda dan media *player* lainnya.

Streaming interaktif adalah hal yang terus menjadi perhatian dan dikembangkan oleh industri internet. Pasalnya, *streaming* interaktif tersebut meningkatkan keterlibatan pengguna Internet dalam berkomunikasi dengan pengguna internet lainnya. Apa sebenarnya *streaming* interaktif tersebut? Jika kita menggunakan teknologi *Internet broadcasting*, *unicast* (*on-demand* / *non real time*) maupun *multicast* (*live* / *real time*), kita hanya dapat duduk, menyaksikan dan mendengarkan saja di depan monitor kita. *Internet broadcasting* sama seperti apabila kita menonton TV atau mendengarkan radio yang merupakan komunikasi satu arah.

Dengan streaming interaktif, kedua belah pihak dapat sama-sama menerima dan mengirimkan informasi pada saat yang bersamaan (*real time* dan *live*) tanpa harus disimpan dahulu ke media penyimpanan atau dibawa ke *streaming server*. Kedua belah pihak yang melakukan *streaming* interaktif tersebut tidak harus sama-sama menggunakan gambar dan suara seperti *Internet video conference*. Bisa jadi hanya satu pihak yang menyiarkan gambar dan suara, sedangkan pihak yang lainnya hanya merespon atau menjawab melalui suara ataupun teks biasa yang diketikkan. *Software* semisal ICQ dan *Yahoo! Messenger* adalah aplikasi *chat* yang berbasis pada teks *streaming*.

Streaming interaktif sendiri terdiri atas 2 jenis, yaitu *two-way* dan *multi-directional*. Untuk *two-way* adalah interaksi 1-*to*-1 melalui Internet. Pasca kejadian teroris 11 September 2001 di New York dan Washington, banyak para pebisnis yang memasang *Internet video conference* di kantornya untuk menyelenggarakan rapat dengan kantor lainnya di tempat yang jauh terpisah. Hal tersebut tentunya merupakan penghematan waktu dan bi-

aya perjalanan. Mark Plus Indonesia juga kerap melakukan *Internet video conference* dengan pakar marketing di luar negeri ketika mengadakan seminar-seminar marketing.

Sedangkan *multi-directional*, adalah interaksi *one-to-many* atau *many-to-many*. Contohnya adalah semisal tayangan debat politik melalui *Internet broadcasting* yang memungkinkan para pemirsanya pada saat itu secara langsung dapat mengajukan pertanyaan melalui audio ataupun teks. Ketika peluncuran situs *Liputan6.com* beberapa tahun silam, diadakan diskusi mengenai media massa berbasis Internet yang menghadirkan beberapa pemimpin redaksi. Diskusi tersebut di-*streaming*-kan dan disiarkan secara *live* dan *real-time* melalui situs *Liputan6.com* tersebut. Para pemirsa diskusi tersebut dapat mengajukan pertanyaan maupun komentar kepada para nara sumber diskusi tersebut melalui teks di *chat room* yang disediakan oleh pihak SCTV.

Hingga kini setidaknya ada tiga jenis format *streaming* yang banyak digunakan di situs-situs Internet. Format tersebut adalah keluaran *Real Media* (.rm / .ra / .ram), *Windows Media* (.asf / .wmf / .asx) dan *QuickTime* (.mov). Masing-masing format tersebut memiliki kekurangan dan kelebihan sendiri-sendiri. Format *Real Media* dan *Windows Media* sangat handal di proses *streaming*, tetapi tidak terlalu handal untuk proses editing dan *playback* lokal. Sedangkan format *QuickTime* rata-rata cukup handal untuk proses *streaming*, editing dan *playback* lokal. Untuk diketahui, *QuickTime* merupakan format *streaming* yang paling lama umurnya, yaitu sejak 1991.

Ketiga format tersebut membutuhkan semacam *player* atau *plug-in* yang terinstal di komputer *client* agar dapat menikmati *streaming* yang ditawarkan suatu situs. Tentu saja, ketiga format tersebut tidak saling kompatibel dengan *player* yang bukan peruntukkannya. Biasanya di sebuah komputer akan terinstal tiga *player* sekaligus, karena setiap situs di Internet belum tentu memilih format *streaming* yang digunakannya. Untuk mendapatkan *player* dari *Real Player*, bisa di *download* pada *http://www.real.com/player*, *Windows Media* dan *http:// www. windowsmedia.com,*.

Codec adalah kependekan dari *compression/decompression*. *Codec*, dalam konteks *streaming*, adalah suatu metode atau algoritma yang terdapat pada sebuah *streaming player* yang fungsinya adalah untuk melakukan proses pengkompresan dan pengdekompresan *file* media *streaming*. Bayangkanlah sebuah *file* media (audio atau video) bagaikan sepotong roti. Volume roti tersebut tentunya banyak memakan tempat, karena ciri fisik dari roti tersebut tidaklah padat.

Jika kita meremas roti tersebut, maka berat atau isi dari roti tersebut tetaplah sama, tetapi volumenya telah banyak berkurang. Fungsi *codec* pada *file* media tidak jauh berbeda dengan proses peremasan pada roti tersebut. *codec* meremas (mengkompresi) *file* media tersebut agar ukurannya dapat diperkecil, lalu *file* tersebut di-*streaming* dan di-*broadcast* melalui Internet. Setelah sampai ke komputer *client*, *file* tersebut kemudian didekompres ke ukuran asal untuk dapat didengarkan atau ditonton. Proses ini memungkinkan kita untuk dapat menikmati media *streaming* dengan lebih cepat (Evdemon, J. 2001).

Ilmu *codec* adalah sebuah seni digital. Banyak hal yang harus dipertimbangkan jika kita ingin melakukan proses kompresi-dekompresi *file* media. Sekedar contoh, semakin besar *file* media, maka akan semakin besar pula ukuran *file* tersebut. Semakin banyak melakukan kompresi *file* tersebut, maka akan semakin berkurang pula kualitas *file* tersebut ketika dinikmati kembali. Untuk *file* video, semakin sedikit kecil kita menentukan *frame* per *second* (fps), maka gambar yang dihasilkan akan patah-patah. Untuk mengatasi hal tersebut, maka akan sering temui bahwa ukuran layar video *streaming* relatif lebih kecil ketimbang ukuran monitor (Passani, L. 2000).

Untuk mengatasi permasalahan tersebut dapat dilakukan dengan cara kompresi, yang dilakukan dengan kualitas video sehingga gambar yang dihasilkan tidak patah-patah. Sekedar informasi, *Windows* media menggunakan varian dari MP4 *Codec*, *Real Network* menggunakan *Intel based Codec* dan *QuickTime*

menggunakan *Sorenson based Codec*. Masih banyak *Codec* lain yang dimiliki dan digunakan oleh pihak-pihak tertentu. Pada Agustus 1999, kode MPEG-4 (MP4) menjadi *open source*. MP4 tersebut pada awalnya digunakan untuk melakukan *ripping* (pengkopian) DVD, karena merupakan *Codec* yang fleksibel dengan teknik kompresi yang handal.

Bagi dunia percaturan *streaming*, *bandwidth* merupakan raja yang memegang peran kunci. Pasalnya, agar sebuah *file* media yang di-*streaming* dan di-*broadcast* dapat kita nikmati sebagaimana mestinya, akses Internet kita haruslah memiliki *bandwidth* yang memadai. Bukan sekedar dari tipe akses yang kita gunakan, apakah 56,6 Kbps, *leased line* ataupun *Internet cable*, tapi juga *bandwidth* antara *Internet Service Provider* (ISP) kita ke *server streaming* yang kita tuju. Anggaplah ukuran awal sebuah *file* video per *frame*-nya adalah sebesar 75 Kb, *file* tersebut berhasil dikompresi hingga 25 Kb. Jika akses Internet kita hanya 56,6 Kbps, maka kita hanya dapat menikmati 2 *frame* per detik (*frames* per *second* - fps). Tetapi 56,6 Kbps itu adalah kondisi ideal. Seringkali yang kita dapatkan ternyata di bawah itu. Sehingga kita hanya bisa menikmati 1 fps saja. 1 fps dengan kualitas yang sudah direduksi sebesar 50 Kb, mungkin tidak akan menghasilkan kualitas gambar yang kita harapkan. Itulah mengapa dalam bisnis *streaming*, *bandwidth* adalah raja. Dari sisi kita pengguna Internet, tidak mudah mendapatkan kepuasan menikmati *streaming* yang berkualitas apabila *bandwidth* dan kecepatan akses internet yang kecil.

Pada akhir abad 19 Heinrich Rudolf Hertz, Nicola Tesla, Alexander Popov, Eduard brandly, Oliver Lodge, Guglielmo Marconi, Adolphus Slaby, dan beberapa insinyur lainnya melakukan percobaan untuk memancarkan dan menerima gelombang elektromagnetik. Pada tahun 1898 Tesla mendemonstrasikan perahu yang dikontrol radio. Pada tahun yang sama Marconi membangun jaringan telegraf tanpa kabel di Inggris. Kejadian ini dianggap sebagai kelahiran radio komunikasi.

Perkembangan sistem *selular* dimulai pada tahun 1970 dimana Ericson memperkenalkan sistem NMT *(Nordic Mobile Telephone)* dan AT&T Bell Laboratories memperkenalkan AMPS (*Advanced Mobile Phone Service*). Pada tahun 1982, *Conference of European Postal dan Telecomunications Administration* (CEPT) mendirikan GSM untuk membuat standar *selular* di Eropa.

Pada tahun 1988 CTIA *(Celluler Telecommunications Industry Association*) membutuhkan suatu sistem *selular* baru untuk mengantisipasi peningkatan jumlah pelanggan *selular*. Setelah melakukan pengembangan selama dua tahun maka ditetapkan standar IS-54 yang dikenal sebagai digital AMPS.

Digital AMPS juga dirasakan kurang memenuhi kebutuhan pelanggan, maka dikembangkan suatu sistem *selular* baru yang menggunakan teknologi CDMA yaitu IS-95 pada 1992 oleh Qualcomm. Pemakaian teknologi CDMA ini memberikan keuntungan - keuntungan sebagai berikut (Santoso, G. 2004):

1. peningkatan kapasitas sistem
2. peningkatan kualitas suara
3. bersifat lebih pribadi dan aman
4. perencanaan sistem menjadi lebih sederhana karena tidak diperlukan perencanaan frekuensi yang kritis sehingga dapat menekan biaya
5. daya pancar lebih kecil sehingga waktu pemakaian baterai menjadi lebih lama dan lebih aman untuk kesehatan pemakai
6. interferensi dengan peralatan elektronik lain lebih kecil.

Dalam pembuatan aplikasi ini perangkat keras atau *hardware* yang digunakan, adalah:

1. Processor : Intel 2.26 Gbyte,
2. Memory : 512 Mbyte,
3. Kartu VGA : 32 Mbyte,
4. CD-Room : Samsung 52X,
5. Monitor, Keyboard, Mouse,Printer.
6. telepon selular CDMA (mendukung WAP)
7. Speaker aktif

Sistem pembelajaran elektronik (m-*learning*) berbasis multimedia menggunakan selular CDMA memerlukan beberapa kriteria perangkat lunak yang a-

kan digunakan agar sistem berjalan dengan baik, yaitu:

1. Sistem Operasi Windows Xp service pack I,
2. Web Server (PHPTriad),
3. Oven wave simulator.
4. Makromedia Dreamweaver MX.
5. Editor Notepad.
6. Adobe Photoshop.
7. Apollo 3gp video converter.
8. *Server streaming*.

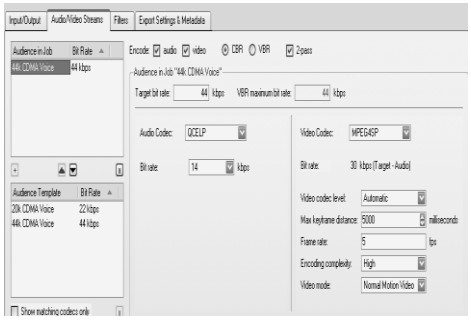
Dalam pengaplikasian media *streaming* pada m-learning dengan menggunakan *selular* CDMA menggunakan bahasa pemrograman WML untuk membangun aplikasi WAP (*Wireless Application Protocol*), dan bahasa *scripting* PHP sebagai *script* untuk penghubung.

PEMBAHASAAN

Dalam membuat sebuah *file streaming* maka dibutuhkan *file* sumber. *File* sumber ini bisa berupa *file* yang secara *live* atau *on-demand*. Setelah dipilih *file video* atau *audio* yang akan di-*streaming*-kan, selanjutnya adalah masuk dalam tahap *encoder*. Seperti yang telah dijelaskan pada Bab II bahwa *encoder* adalah merupakan inti dari proses *streaming*. Pada bagian *encoder*, sebenarnya terdapat banyak software yang digunakan antara lain: windows media menggunakan MP4 *Codec*, *Real Network* menggunakan *Intel based Codec* dan *QuickTime* menggunakan *Sorenson based Codec*. Pada perancangan media *streaming* ini menggunakan *video codec* MPEG4 dan keluaran dari hasil *codec* tersebut ber-*type file* .3pp untuk *streaming* yang dilakukan secara *on-demand* dan jika ingin melakukan *streaming* secara *live* maka *type file* yang dihasilkan adalah .sdp.

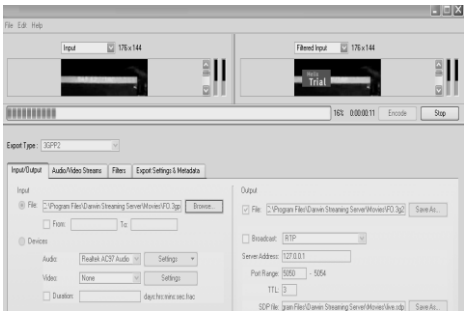
Dalam melakukan proses *codec* ini, menggunakan *software* dari realnetwork (helix). Sebelum menentukan sumbernya, terlebih dahulu dilakukan seting-an dasar agar *file* tersebut bisa disesuaikan dengan tampilan pada ponsel. Tab pada *audio/video stream,* pada bagian *audio codec* sebaiknya tidak usah dirubah. Pada bagian *video codec* dipilih MPEG4, pada *frame rate* disesuaikan dengan kebutuhan (frame rate yang dianjurkan adalah 7fps – 25 fps).

Selanjutnya adalah untuk mengatur *output* ukuran *frame* dari file tersebut agar bisa sesuai dengan tampilan ponsel. Caranya adalah: tab pada bagian *filters* dan beri tanda centang pada *resize* dan pilih QCIF, dipilih QCIF karena ini sudah distandarkan oleh CCITT (*Consultative Commite for International Telephone and Telegraph*) untuk standar *videocoferencing* dan aplikasi *video*, yang kemudian dikenal dengan nama H.261.



Gambar 6. Pengaturan *audio/video stream*

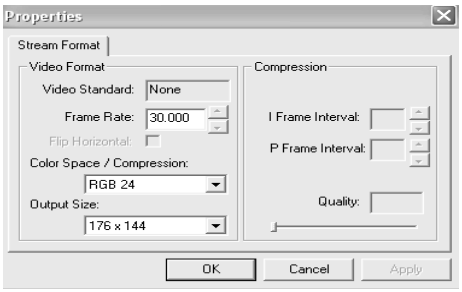
Proses pengkompresan file *video* secara *on-demand* dapat dilakukan dengan cara: pertama tama masukan file video yang akan dikompresi dengan cara tab pada *input/output*, kemudian tekan tombol *browse* pada bagian *input*. Setelah file *video* berhasil dimasukan kemudian pada bagian output berikan tanda centang pada bagian *file*, kemudian tekan tombol *save as* untuk menyimpan *file* hasil kompresi. Setelah itu tekan tombol *encode.*



Gambar 7. Proses kompresi secara *on demand*

Berikutnya adalah proses kompresi secara *live*, tab pada input/output

kemudian pada bagian input pilih pada bagian *device*. Selanjutnya adalah mengatur sumber yang akan dikoneksikan, untuk pengaturan pada audio bisa digunakan pengaturan standart. Tapi pada *video* tidak bisa jika menggunakan pengaturan standar maka keluaran dari *video* akan besar. Untuk melakukan pengaturan *video* pilih *settings* kemudian pilih *video capture pin*. Sehingga akan muncul gambar sama seperti Gambar 8, kemudian ganti *color space/compresion* dengan menggunakan RGB 24 dan selanjutnya adalah mengganti *output size* dengan ukuran 176 x 144. Pada bagian *output* centang pada *broadcast*, selanjutnya simpan file tersebut pada server streaming dengan ekstensi .sdp.



Gambar 8. Pengaturan *Video*

Selanjutnya adalah menyimpan *file streaming* tersebut pada streaming server. *Streaming server* yang bisa digunakan untuk *protocol* rtsp adalah *streaming server* buatan realnetwork dan buatan dari apple. Prinsip kerja dari kedua *streaming server* tersebut sebenarnya hampir sama. Hanya saja yang membedakan adalah OS (*operation system*) dari komputer, untuk *server* buatan realnetwoork lebih dikhususkan untuk OS pada linux, sedangkan *server* buatan apple lebih dikhususkan untuk produknya sendiri yaitu apple. Kedua *server streaming* tersebut bisa berjalan juga pada windows.

Fungsi dari *streaming server* ini sebenarnya hanya lebih difokuskan untuk *protocol* saja. Penggunaan *server* secara khusus dikarenakan pada WAP tidak bisa memutar file streaming jika server yang digunakan tidak memiliki server streaming, dalam hal ini mendukung *protocol* rtsp.

Pada sub pokok bahasan ini yang diamati adalah tentang streaming. Pengamatan yang dilakukan adalah pengamatan terhadap protokol rtsp dan mencoba menggunakan pada berbagai operator. *Web* yang digunakan untuk pengamatan adalah http://wap.abgso.com. Media yang diamati adalah tampilan rtsp pada tampilan PC dan pada tampilan *ponsel*.

Operator Starone (jagoan)

Pada pengujian ini, yang dipakai a-dalah parameter tampilan *web* yang tam-pil pada *ponsel.* Pada percobaan pada tampilan *mini browser ponsel web* ter-sebut tidak dapat dibuka, dengan menun-jukan keterangan *error*: *network down*. Percobaan tersebut telah dilakukan selama 2 minggu berturut, dengan waktu yang berbeda, membuka alamat WAP yang berbeda-beda dan selalu menampilkan kalimat yang sama. Dari percobaan tersebut dapat ditarik kesimpulan bahwa *gateway* WAP dari jagoan belum diaktifkan.

Sedangkan pada percobaan berikutnya adalah dengan menggunakan kabel data dan menjadikan ponsel CDMA sebagai *modem*. Dari percobaan tersebut koneksi berhasil dilakukan, dan langsung dicoba untuk membuka file *streaming*-nya. Pada saat membuka file streaming dengan menggunakan protocol rtsp, maka secara langsung real player langsung membuka file tersebut. Percobaan tersebut dilakukan lagi pada PC yang tidak mempunyai real player maka pada tampilan PC tersebut langsung memunculkan pesan *error*. Dari percobaan ini dapat disimpulkan bahwa, protokol rtsp jika ingin ditampilkan pada PC maka didalam PC telah ter-*instal* real player.

Gambar 9. Tampilan protocol RTSP pada PC

Operator Fren

Percobaan yang berikutnya dilakukan adalah mengganti operator (menggunakan operator fren). Percobaan yang pertama kali dilakukan adalah menghubungkan *ponsel* sebagai modem, dan membuka website yang sama. Percobaan analisa dilakukan dengan langsung membuka file streaming, maka langsung menghasilkan tampilan yang sama dengan operator jagoan (Gambar 9).

Percobaan yang berikutnya adalah dengan membuka tampilan *web* pada ponsel. Setelah berhasil terhubung dengan wap, langsung dilakukan pengujian file *streaming*-nya*. File streaming* tersebut berhasil tampil yang ditunjukan pada Gambar 10.

(a) (b)

Gambar 10. *Streaming* pada *ponsel* CDMA

Operator flexi (PT. Telkom)

Untuk tampilan pada PC, masih menghasilkan tampilan yang sama, dimana langsung diambil alih oleh realplayer.

Analisa yang berikutnya adalah dengan menggunakan *ponsel*. Pada saat dicoba untuk masuk pada *gateway* WAP, sukses dan berhasil masuk pada website wap.abgso.com. Permasalahan yang timbul adalah pada saat mengakses file *streaming,* pada saat membuka koneksi pada rtsp. Percobaan tersebut dilakukan selama 1 minggu, dan selalu menampilkan isi yang sama (isi yang ditampilkan: *no response try again*). Atas permasalahan ini penulis mencoba menghubungi operator telkom, dan operator tersebut mengatakan pada saat ini flexi masih belum mendukung streaming. Setelah dianalisa permasalahan ini ditimbulkan karena rtsp melakukan koneksi dengan menggunakan alamat IP (*Internet Protocol*) dari *server*. Dari permasalahan ini dapat diambil kesimpulan bahwa ISP (*Internet Service Provider*) dari operator telkom masih belum melakukan kerjasama dengan pemilik *server*, sehingga *file streaming* tersebut tidak dapat diakses.

KESIMPULAN

Dalam membuat sebuah web untuk memutar *file streaming* harus menggunakan *server* yang telah mendukung *streaming*.

File streaming yang dikhususkan untuk diputar pada *handphone* memakai *protokol* rtsp (*real time streaming protocol*).

Dalam membuat *file* video *streaming* agar hasil tetap bagus, sebaiknya *file* video *streaming* tersebut dirubah ukuran atau *frame*-nya agar lebih kecil sehingga ukuran *file* tersebut tidak terlalu besar.

Dalam pemilihan *hosting* sebaiknya memakai *hosting* yang mempunyai *bandwidth* agak besar sehingga *file streaming* tersebut dapat diputar tanpa terjadinya kehilangan data.

Dalam merancang tampilan *website* untuk tampilan *handphone* sebaiknya dirancang agar ukuran *file*-nya tidak besar, karena dapat mempengaruhi waktu akses dan biaya.

DAFTAR PUSTAKA

Ariwibowo, A. 2003. *Multimedia dan Streaming dengan Snychronized Multimedia Integration Language.* Jakarta.

Daryanto, T. 2005. *Sistem Multimedi dan Aplikasinya*. Yogyakarta.

Evdemon, J. 2001. *XML dan WAP*, Chieft Architect, XML Solutions. http://www.eccnet.eccnet.com/pub/dc-xmlug/Evdemon-WAP.pdf

Passani, L. 2000. *Creating WAP Service.* http://wwwddj.com/articles/2000/0007/ 0007toc.htm

Santoso, G. 2004. *Sistem selular CDMA.* Yogyakarta. (?)