BAB II

LANDASAN TEORI

Pembuatan Layanan Informasi dan Isi Ulang Pulsa Super menggunakan HTTP *Connection* lewat GPRS, PHP digunakan untuk pengolahan dan manajeman layanan informasi pada server Pulsa Super dimana data layanan dan informasi tersebut disimpan pada sebuah database. Database server yang digunakan adalah MySQL dan untuk web server digunakan Apache. Sedangkan untuk aplikasi pada telepon seluler dibuat dengan menggunakan *platform* J2ME dengan menggunakan J2ME Wireless Toolkit sebagai *compiler*. Untuk pengaksesan informasi layanan dari telepon seluler ke web server menggunakan HTTP *Connection* lewat GPRS. Jadi pada bab ini dibahas mengenai dasar teori dari Sistem Informasi, GPRS, J2ME, Http Connection, Web Server Apache, PHP, dan database server MySQL yang digunakan untuk menunjang pembuatan sistem secara keseluruhan.

II.1 Sistem Informasi

II.1.1 Definisi Sistem Informasi

Sistem informasi mengumpulkan, memproses, menyimpan, menganalisa, dan menyebarkan informasi untuk maksud tertentu.

Sistem informasi terdiri dari: *inputs* (data dan perintah-perintah), *outputs* (reports dan perhitungan-perhitungan), mekanisme *feedback* yang mengontrol proses pengerjaan dan *environment* kerja sistem.

II.1.2 Komponen Sistem Informasi

Komponen dari Sistem Informasi adalah sebagai berikut:

1. Hardware adalah seperangkat peralatan yang terdiri dari *processor*, *monitor*, *keyboard* dan *printer*.

2. Software adalah kumpulan program yang memungkinkan *hardware* untuk memproses data.

3. *Database* adalah kumpulan file dan tabel yang saling berelasi satu sama lain untuk menyimpan data.

4. *Network* adalah penghubung sistem yang memungkinkan berbagi pakai *resource* diantara komputer-kompter yang saling berhubungan.

5. Prosedur adalah seperangkat instruksi mengenai bagaimana mengkombinasikan komponen-komponen yang telah disebutkan di atas.

6. Pengguna adalah orang-orang yang bekerja dengan sistem atau yang menggunakan keluaran sistem.

II.1.3 Komponen Sistem Informasi Berbasis Mobile

Dalam membangun sistem informasi, kebutuhan akan perangkat lunak sangat vital disamping kebutuhan akan perangkat keras. Perangkat lunak yang dibangun harus tepat sasaran dan mampu meningkatkan kinerja sistem

secara keseluruhan. Saat ini perkembangan sistem informasi dan proses penggunaannya tidak berhenti pada perangkat lunak berbasis web dan penggunaan PC (Personal Computer) dalam mengakses sistem informasi. Evolusi masih terus berlanjut dan berbagai teknologi baik perangkat lunak maupun perangkat keras dalam mendukung sistem informasi terus berkembang.

Dewasa ini perkembangan berbagai perangkat *mobile* seperti telepon seluler sangat mendominasi pasar perangkat keras. Ini disebabkan kemudahan orang dalam membawa perangkat ini dengan nyaman (*portable*) dan kemampuan komputasinya yang semakin lama semakin cepat. Beberapa tahun lalu, berbagai perangkat *mobile* (*handheld*) diklasifasikan menjadi beberapa macam misalnya *smartphone*, PDA, *comunicator*, dan sebagainya. Namun pada pengembangannya pengklasifikasin tersebut mengkerucut karena fungsionalitasnya hampir tidak ada bedannya diantara perangkat-perangkat handheld yang baru, sehingga lebih tepat jika disebut *embedded system*, dimana satu perangkat *handheld* mampu menampung semua fungsi-fungsi bisnis *end user* (*all in one*).

Sistem informasi yang diakses melalui PC misalnya dengan menggunakan web browser (Internet Explorer, Mozzila) melalui protokol HTTP dapat dengan mudah diakses, tidak demikian halnya bila orang mengakses melalui pengkat mobile semisal telepon seluler. Selain adanya kendala perangkat memori yang ada pada perangkat mobile, infrastruktur jaringan internet tidak menyatu dengan jaringan perangkat *mobile* (dalam hal ini jaringan telepon seluler). Untuk mengatasi kendala tersebut dunia perangkat lunak juga terus berkembang

seiring perkembangan perangkat keras. Untuk mengakses informasi berbasis mobile melalui jaringan internet salah satunya dapat menggunakan HTTP *Connection* melalui teknologi GPRS. Data dan layanan yang disediakan diletakkan pada server tertentu, sehingga pemrosesan data dan layanan terjadi di sisi server dan kemudian hasilnya dapat akses oleh perangkat *mobiile*. Pada Tugas akhir kali ini alur informasi berawal dari permintaan atau *request* melalui aplikasi pada telepon selelur dalam hal ini member pulsa super, yang kemudian direspon dan di proses oleh server Pulsa Super dan hasil pemrosesan diterima oleh member melalui aplikasi pada telepon seluler juga. Dengan sistem ini maka informasi yang didapat bersifat langsung (*real time*) sehingga dapat diproses secara cepat, tepat dan efisien.

II.2 GPRS

II.2.1 Sekilas Tentang GPRS

GPRS yang termasuk dalam kelas 2.5 G merupakan standard komunikasi data di jaringan GSM yang kecepatan transfernya mencapai 115 kbps. Dengan adanya GPRS ini jaringan GSM bisa memisah paket data kecepatan tinggi dengan suara. Dengan adanya GPRS ini pengguna bisa terus terkoneksi ke internet. Pengguna tidak perlu dial up terus menerus ketika akan melakukan koneksi ke internet. Dengan menggunakan media GPRS ini, biaya internet dihitung berdasarkan banyaknya data yang dikirim/diterima.

GPRS disebut teknologi 2.5 G karena merupakan langkah awal menuju teknologi transfer data kecepatan tinggi lewat jaringan nirkabel (3G). Sehingga sering disebut-sebut sebagai teknologi kunci untuk data bergerak. Secara rinci ada beberapa faktor yang menjadi pertimbangan bahwa GPRS merupakan teknologi kunci untuk data bergerak, yakni;

- mampu memanfaatkan kemampuan cakupan global yang dimiliki GSM (2G).
- memperkaya utiliti investasi untuk perangkat GSM yang sudah ada.
- merupakan teknologi jembatan yang bagus menuju generasi ke 3.
- berbasis paket data yang lebih efiesien dalam penggunaan sumber daya.
- memiliki laju data sampai 115 kbps yang berarti dua kali lipat daripada koneksi 'dial up' yaitu 56 kbps.

Dengan adanya GPRS ini operator GSM dapat menambah layanan bagi para pengguna. Pengguna tidak hanya bisa melakukan komunikasi suara namun juga bisa melakukan komunikasi data. Beberapa layanan yang berkembang dengan adanya jaringan GRPS ini antara lain:

- MMS *(Multimedia Messaging System)*, dengan MMS ini pengguna bisa mengirimkan pesan dalam bentuk multimedia (suara, klip video, gambar).

- *Traffic Monitoring*, dengan layanan ini pengguna bisa melihat keadaan lalu lintas di suatu tempat seacara *real time*, dengan maksud agar mengetahui daerah mana yang lalu lintasnya padat dan daerah mana yang lalu lintasnya sepi.

- VOIP *(Voice Over IP)*, layanan ini biasanya digunakan antar pengguna PDA.
  Pemakai PDA pertama harus menginstal suatu program terlebih dahulu baru bisa menggunakan VOIP. Teknologi ini akan efektif bila tarif GPRS dihitung secara *flat*, sehingga walaupun banyak data yang ditransfer namun harga yang dibayarkan tetap sama.

II.2.2 Keunggulan GPRS

GPRS memiliki beberapa keunggulan antara lain :

- Mobilitas
  Kemampuan untuk mempertahankan suara yang konstan pada saat bergerak. GPRS memberikan peningkatan dalam kualitas layanan data yang diberikan, dilihat dari keutuhan data, waktu yang dibutuhkan, dan fitur lain yang didukung oleh GPRS.

- Always on
  Pelanggan dapat memperoleh konektivitas pada saat diperlukan. Jadi pelanggan tidak perlu melakukan *dial up* ke server.

- Kecepatan tinggi
  GPRS menggunakan 1 sampai 8 *timeslot* dari *channel* radio yang dapat memberikan kecepatan lebih dari 115 kBps.

- Biaya penggunaan GPRS yang lebih murah
  Pengguna GPRS dikenai biaya berdasarkan jumlah data yang diterima, tidak berdasarkan lamanya waktu pengguna terkoneksi ke jaringan. Pada umumnya tarif GPRS di Indonesia adalah Rp. 5/kb.

- Beban terhadap j aringan relatif ringan
  Dengan GPRS, pengguna dapat terkoneksi terus, tanpa mengganggu kepadatan lalu-lintas jaringan. Para pengguna GPRS dapat menggunakan sebuah *channel* radio secara bersamaan dan pengguna tidak membebani jaringan pada saat tidak ada transfer data.

II.2.3 Aplikasi GPRS

Dengan GPRS memungkinkan munculnya berbagai aplikasi baru untuk pelanggan jaringan *wireless.* Dengan adanya penggabungan karakteristik maka memberikan gambaran luas pada aplikasi yang mungkin ditawarkan ke pelanggan jaringan *wireless.* Pada umumnya, aplikasi

dapat dipisahkan menjadi dua kategori, yaitu: aplikasi untuk korporasi dan pribadi, antara lain:

- Komunikasi - e-mail, fax, akses Intemet.
- Value Added Services - layanan informasi, permainan.
- E-Commerce - retail, pembayaran tiket, *e-banking, financial trading.*
- Location Based Applications - jadwal penerbangan, pencari lokasi.
- Periklanan

II.2.4 Terminal GPRS

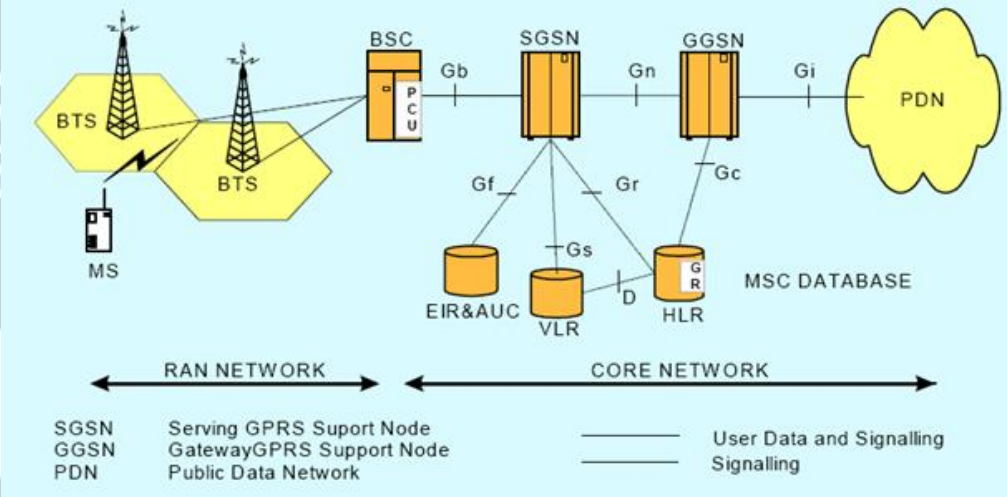
Terminal adalah macam-macam dari ponsel atau *Mobile Station* (MS) yang digunakan oleh pengguna dalam mengakses layanan GPRS. Terminal GPRS dapat dibagi menjadi tiga kelas: A, B, atau C.

- Terminal A *class-A* adalah semua MS yang mendukung GPRS dan layanan lain seperti SMS dan suara. Kelas ini mendukung pengiriman dan penerimaan data secara bersamaan. Jadi pada kelas ini memungkinkan untuk mengadakan dan menerima panggilan dalam dua layanan secara bersamaan.

- Terminal A *class-B* hampir sama dengan terminal A *class-A.* Perbedaannya pada terminal ini, MS tidak mendukung pengiriman dan penerimaan data secara bersamaan. Jadi pengguna MS ini hanya dapat mengadakan atau menerima panggilan secara bergantian, tidak secara bersamaan.

- Terminal A *class-C* adalah MS yang tidak mendukung pengiriman dan penerimaan data secara bersamaan. Pengguna MS ini harus memilih layanan mana yang akan dipakainya. Oleh karena itu, terminal ini hanya dapat mengadakan dan menerima panggilan dengan layanan yang dipilihnya atau layanan *default* dari MS.

II.2.5 Arsitektur Jaringan GPRS



Gambar 2.1 Arsitektur Jaringan GPRS

Pembangunan arsitektur jaringan GPRS dapat menggunakan elemen jaringan GSM yang masih memungkinkan. Elemen tersebut harus disesuaikan atau dimodifikasi sedemikian rupa supaya dapat berfungsi secara optimal pada jaringan GPRS. Penyesuaian biasanya dilakukan dengan memperbarui *software* yang ada. Selain itu penambahan elemen jaringan baru juga dibutuhkan untuk membangun paket data yang efektif. Dari gambar 2.1 dapat dilihat bahwa jaringan GPRS terdiri dari empat elemen utama yaitu:

a. Subscriber Terminal

Terminal adalah jenis dari ponsel atau MS yang digunakan oleh pengguna dalam mengakses layanan GPRS. Terminal baru (TE) dibutuhkan karena terminal lama sudah tidak dapat lagi menangani hubungan jaringan yang ditingkatkan. Masalah utama yang terdapat pada terminal lama adalah terminal lama tidak mempunyai kemampuan untuk mempaketkan lalu-lintas data secara langsung. Pada saat ini banyak terminal yang sudah mendukung fasilitas GPRS.

b. BSS (Base Station System)

BSS yang biasa disebut dengan BSC *(Base Station Controller)* digunakan untuk mengenali dan memberikan data pelanggan ke jaringan yang sedang melayani airea tersebut. Pada BSS dibutuhkan penambahan *software* agar dapat digunakan secara optimal pada jaringan GPRS. Selain itu pada tiap BSS juga membutuhkan penambahan *hardware* yaitu: PCU *(Packet Control Unit)*. PCU digunakan untuk mengatur pengiriman paket data antara perangkat pengguna dan jaringan GPRS. PCU juga mendukung pengiriman *frame* data pada protokol GPRS.

c. Core Network

GPRS menambahkan dua elemen baru pada jaringannya untuk dapat menangani paket data secara efektif. Dua elemen tersebut adalah:

1. SGSN (Serving GPRS Support Node)

   SGSN dihubungkan secara langsung ke PCU yang terdapat pada BSS (menggunakan koneksii *Frame Relay)*. SGSN berfungsi untuk mengirimkan paket data ke MS di dalam area layanannya, mengirimkan

query ke HLR *{Home Location Registers)* untuk mendapatkan data profil dari pelanggan GPRS. Jika SGSN mendapatkan MS baru yang diberikan di dalam area layanannya maka SGSN akan mengolah proses registrasi dari pelanggan baru dan menyimpan data pelanggan ke dalam database menurut area layanan yang diberikan.

2. GGSN (Gateway GPRS Support Node)

   GPRS berfungsi sebagai gateway pada jaringan data ekstemal dan menyediakan layanan seperti menjaga keamanan pada akses jaringan ekstemal. Yang termasuk jaringan ekstemal adalah Internet, *Private Intranets,* dam jaringan X.25. GGSN digunakan sebagai *interface* ke jaringan IP eksternal seperti Intemet, layanan GPRS lain, *Enterprise Intranet.* Fungsi lainnya adalah melindungi jaringan, pelanggan, dan pemetaan alamat. Satu atau lebih GGSN mungkin digimakan untuk mendukung banyak SGSN.

d. Database

Database yang digunakan pada jaringan GPRS secara umum dapat dibagi menjadi dua yaitu:

1. HLR (Home Location Register)

   HLR digunakan untuk menyimpan data profil dari pelanggan GPRS, sehingga SGSN dapat memperoleh informasi data profil dari pelanggan pada HLR.

2. VLR (Visitor Location Register)

   VLR digunakan untuk menyimpan data profil dari MS yang bergerak dari satu area ke area yang lain.

II.2.6 Cara Kerja Jarmgan GPRS

Pada saat fitur GPRS dijalankan rnaka secara otomatis perangkat tersebut mencari *channel* GPRS. Jika *channel* yang tepat sudah ditemukan, perangkat akan langsung melakukan koneksi ke jaringan. SGSN menerima adanya koneksi baru dari perangkat tersebut. Kemudian SGSN mengirimkan permintaan ke HLR (mendapatkan data profil dari pelanggan GPRS tersebut, dan memeriksanya). Jika benar, maka dibangun koneksi ke jaringan GPRS.

SGSN menggunakan data profil APN *(Access Point Name)* untuk mengenali jaringan dan operator. APN digunakan untuk menentukan GGSN mana yang akan menangani pelanggan tersebut. Gateway yang ditentukan melakukan pengecekan pada pengguna dengan metode RADIUS *(Remote Authentication Dml-ln User Service)* dan mengalokasikan pengguna dengan alamat IP yang dinamik sebelum mengatur koneksi dengam jaringan di luar, termasuk juga pengaturan QoS dan VPN (*Virtual Frivate Network).*

Pada jaringan GPRS, data yang dikirimkan dipecah menjadi paket-paket kecil, kemudian paket-paket ini yang dikirimkan, ketika paket-paket tersebut datang di tempat penerima, maka di susun kembali menjadi sebuah data utuh.

II.3 Java

II.3.1 Sekilas tentang Java

Java adalah bahasa pemrograman yang disusun oleh James Gosling yang dibantu oleh rekan-rekannya dalam sebuah *project* bernama *Green* yang meneliti pembuatan bahasa pemrograman yang akan digunakan pada *chip-chip*

embeded untuk device *Intelligent Consumer Electronic* di suatu perusahaan perangkat lunak yang bernama Sun Microsystems pada tahun 1991. Bahasa pemrograman ini mula-mula diinisialisasikan dengan nama Oak, namun pada tahun 1995 diganti namanya menjadi Java.

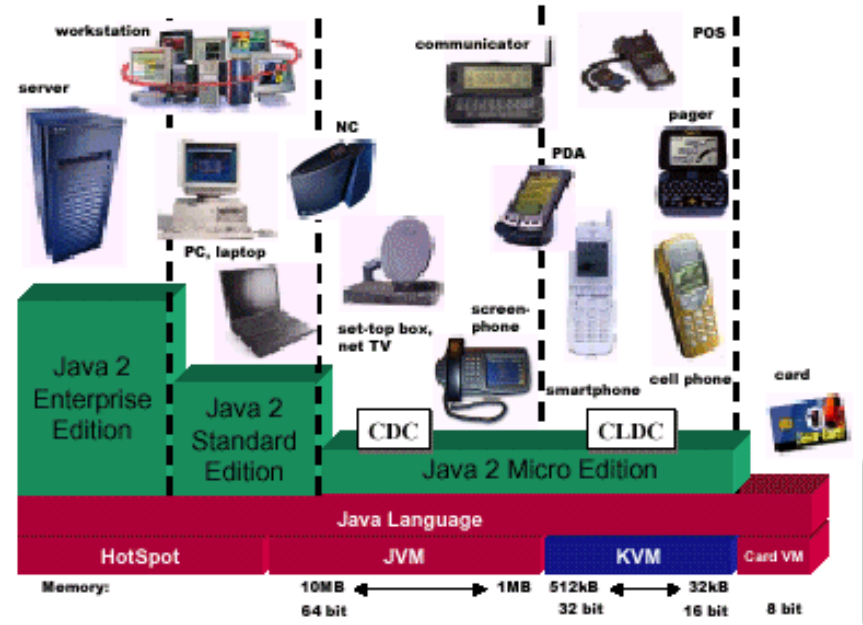
Alasan utama pembentukan bahasa Java adalah untuk membuat aplikasi-aplikasi yang dapat diletakan di berbagai macam perangkat elektronik, sperti *microwave oven* dan *remote control*, sehingga Java harus bersifat *portable* atau yang sering disebut dengan *platform-independent* (tidak tergantung pada *platform*). Itulah yang menyebabkan dalam dunia pemrograman Java, dikenal adanya istilah *write once, run everywhere*, yang berarti kode program hanya ditulis sekali, namun dapat dijalankan dibawah *platform* manapun tanpa harus melakukan perubahan kode program.

Pada awal perilisannya, versi Java masih disebut dengan JDK (*Java Development Kit*). Dalam JDK, semua kebutuhan untuk pengembangan program dan eksekusi program masih tergabung jadi satu. Penaman ini berlaku sampai Java 1.1. Namun sekarang, setelah Java 1.2, Sun Microsystems menamainya dengan JSDK (*Java Software Development Kit*) dalam hal ini kebutuhan untuk pengembangan program dipisahkan dengan kebutuhan eksekusi. Bagian *software* yang digunakan untuk kebutuhan eksekusi program disebut dengan JRE (*Java-Runtime Environment*). Selanjutnya, Java 1.2 disederhanakan penamaannya menjadi Java 2.

Sun Microsystems telah mendefinisikan tiga buah edisi dari Java 2, yaitu sebagai berikut :

1. *Java 2 Standard Edition* (J2SE)
   Digunakan untuk mengembangkan aplikasi-aplikasi *dekstop* dan *applet* (aplikasi Java yang dapat dijalankan di dalam *browser web*).
2. *Java 2 Enterprise Edition* (J2EE)
   Merupakan *superset* dari J2SE yang memperbolehkan kita untuk mengembangkan aplikasi-aplikasi bersekala besar (*enterprise*), yaitu untuk melakukan pembuatan aplikasi-aplikasi di sisi server yang menggunakan EJBs (*Enterprise Java Beans*), aplikasi *web Servlet* dan JSP (*Java Server Page*) dan teknologi lainnya seperti COBRA dan XML (*Extensible Markup Language*).
3. *Java 2 Micro Edition* (J2ME)
   Merupakan *subset* dari J2SE yang digunakan untuk menangani pemrograman di dalam perangkat-perangkat kecil, yang tidak memungkinkan untuk mendukung implementasi J2SE secara penuh.

Namun saat ini selain 3 kategori diatas terdapat juga *Java Card*, kategori khusus yang digunakan untuk mengembangkan aplikasi pada *smart card* seperti aplikasi kartu telepon *chip*, *mobile banking* dan pemrograman SIM *card* pada ponsel. Kemudian ada juga Java *web service*, yaitu aplikasi Java berbasis *enterprise* yang menggunakan standar XML dan protokol tertentu dalam bertukar data dengan klien.



Gambar 2.2 Java Platform

Teknologi Java sesuangguhnya jauh lebih luas dari yang telah digambarkan diatas. Karena pengembangan Aplikasi Layanan Informasi dan Isi Ulang Pulsa Super menggunakan HTTP Connection lewat GPRS ini berjalan pada mobile device, maka pada penjelasan selanjutnya mengulas tentang J2ME.

II.3.2 *Java 2 Micro Edition* (J2ME)

Java 2 Micro Edition atau yang biasa disebut J2ME adalah lingkungan pengembangan yang didesain untuk meletakkan perangkat lunak *Java* pada barang elektronik beserta perangkat pendukungnya. Pada J2ME, jika perangkat lunak berfungsi baik pada sebuah perangkat maka belum tentu juga berfungsi baik pada perangkat lainnya. J2ME membawa java ke dunia informasi, komunikasi, dan perangkat komputasi selain perangkat komputer *dekstop* yang biasanya leih kecil dibandingkan

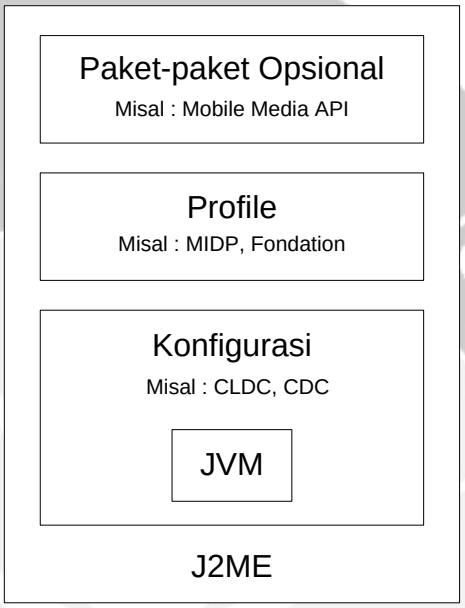
perangkat komputer *dekstop*. J2ME biasa digunakan pada telepon seluler, *pager*, *personal digital assistansts* (PDA's) dan sejenisnya.

J2ME adalah bagian dari J2SE, karena itu tidak semua *library* yang ada pada J2SE daoat digunakan pada J2ME. Tetapi J2ME mempunyai beberapa *library* khusus yang tidak dimiliki J2SE.

Teknologi J2ME juga memiliki beberapa keterbatasan, terutama jika diaplikasikan pada ponsel. J2ME sangat tergantung pada perangkat lunak (device) yang digunakan, bisa dari segi merek ponsel, maupun kemampuan ponsel, dan dukungannya terhadap teknologi J2ME. Misalya, jika sebuah ponsel tidak memiliki kamera maka jelas J2ME pada ponsel tersebut tidak dapat mengakses kamera. Keterbatasan lainnya adalah pada ukuran aplikasi, karena memori pada ponsel sangat terbatas. Sebagian besar ponsel tidak mengijinkan aplikasi J2ME menulis pada file karena alasan keamanan. Saat ini ada 2 jenis aplikasi dari J2ME, yaitu:

1. *Walled Garden Application*, aplikasi ini berdiri sendiri (*stand alone*), berjalan pada ponsel dan tidak mengakses sumber data eksternal melalui jaringan atau *network*.
   Contoh: kalkulator, single player game, *worldclock*.
2. *Network Aware Application*, aplikasi ini berinteraksi dengan jaringan dan mempunyai kemampuan mengakses sumber data eksternal.

J2ME terdiri dari tiga buah komponen yang dikenal dengan istilah *configuration* dan *profile* dan paket-paket opsional, seperti yang ditunjukan oleh gambar 2.3



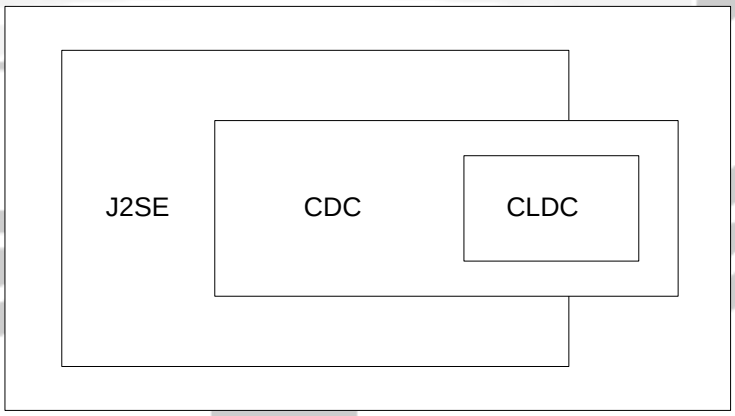
Gambar 2.3 Bagian di dalam platform J2ME

II.3.2.1 J2ME *Configuration*

Configuration merupakan Java *library* minimum dan kapabilitas yang dipunya oleh para pengembang J2ME, yang maksudnya sebuah mobile device dengan kemampuan Java akan dioptimalkan untuk menjadi sesuai. *Configuration* hanyalah mengatur hal-hal tentang kesamaan sehingga dapat dijadikan ukuran kesesuaian antar device, Misalnya sebuah lampu seperda dirancang sedemikian rupa sehingga dapat digunakan oleh berjenis-jenis sepeda. Dalam J2ME telah didefinisikan dua buah konfigurasi yaitu CLDC (*Connected Limited Device Configuration*) untuk perangkat kecil dan CDC (*Connected Device Configuration*) untuk perangkat yang lebih besar.

1. *Connected Limited Device Configuration* (CLDC)

   CLDC atau *Connected Limited Device Configuration* adalah perangkat dasar dari J2ME, spesifikasi dasar yang berupa *library* dan API yang diimplementasikan pada J2ME, seperti yang digunakan pada telepon seluler, pager, dan PDA. Perangkat tersebut dibatasi dengan keterbatasan memori, sumber daya, dan kemampuan memproses. Spesifikasi CLDC pada J2ME adalah spesifikasi minimal dari *package*, kelas, dan sebagian fungsi *Java Virtual Machine* yang dikurangi agar dalam diimplementasikan dengan keterbatasan sumber daya pada alat-alat tersebut, JVM yang digunakan disebut KVM (*Kilobyte Virtual Machine*). Posisi CLDC pada arsitektur J2ME dapat diliat pada gambar 2.4.



Gambar 2.4 Lingkup Configuration

2. Connected Device Configuration (CDC)

   CDC atau *Connected Device Configuration* adalah spesifikasi dari konfigurasi J2ME. CDC merupakan komunitas proses pada Java yang memiliki standarisasi. CDC terdiri dari *virtual machine* dan kumpulan *library* dasar untuk dipergunakan pada

profile industri. Implementasi CDC pada J2ME adalah *source code* yang menyediakan sambuangan dengan macam-macam *platform*. Berikut adalah perbandingan antara CLDC dengan CDC.

CLDC	CDC
Mengimplementasikan subset dari J2SE	Mengimplementasikan seluruh fitur pada J2SE
JVM yang digunakan dikenal dengan nama KVM	JVM yang digunakan dikenal dengan nama CVM
Digunakan pada perankat *handheld* dengan ukuran memori terbatas (160-512 KiloBytes)	Digunakan pada perrangkat *hendeld* dengan ukuran memori minimal 2 Megabytes
Prosesor 16 bit / 32 bit	Prosesor 32 bit

Seperti yang sudah disebutkan bahwa CLDC digunakan untuk implementasi program *Java* pada perangkat-perangkat keras dengan ukuran memori terbatas. Akibatnya, fitur-fitur yang kurang penting untuk diimplementasikan dalam perangkat *handheld* yang bersangkutan dari *Java* 2 harus dibuang. Fitur-fitur yang dibuang tersebut antara lain :

1. Tidak ada dukungan *floating point*, kelas-kelas untuk perhitungan floating point, yaitu *java.lang.float* dan *java.lang.double* dibuang dari CLDC.
2. Tidak ada dukungan untuk finalisasi objek. *Garbage collector* yang digunakan untuk membersihkan memori, membuang fungsi finalize pada kelas *java.lang.object*.
3. Tidak ada dukungan untuk JNI. Kelas JNI yang memungkinkan Java mengakses *libraly* yang dibuat dengan bahasa selain Java, tidak didukung CLDC.

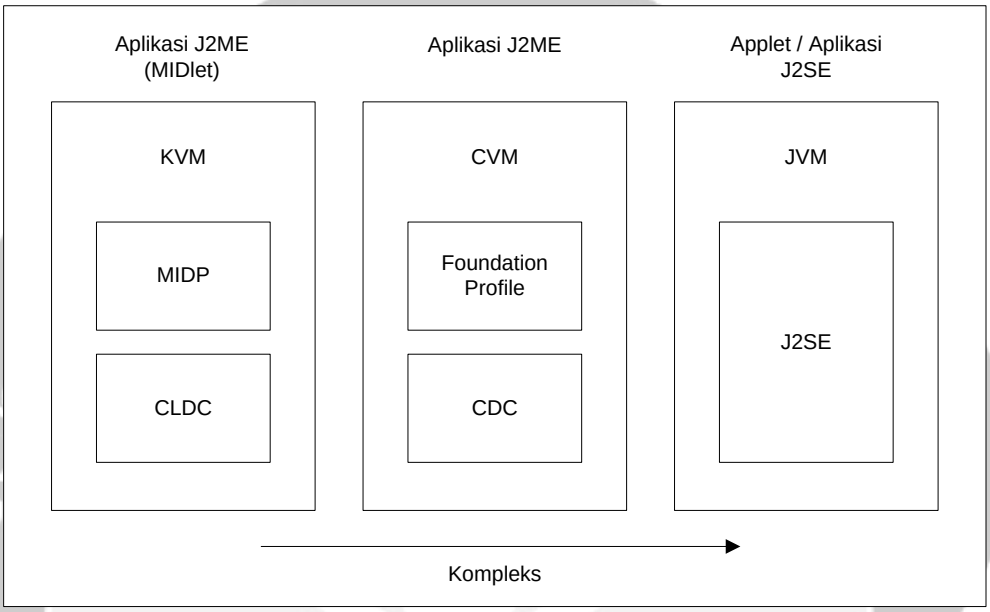
Penanganan eksepsi yang terbatas CLDC hanya mendefinisikan tiga kelas untuk penanganan eksespsi, yaitu java.lang.error, java.lang.outmemory, dan java.lang.VirtualMachineError.

II.3.2.2 J2ME *Profile*

Profile bisa dikatakan sebagai sebuah tambahan untuk *Configuration*. *Profile* menyediakan pustaka untuk para perancang J2ME dalam merancang aplikasi. Sebagai contoh, *Mobile Information Device Profile* (MIDP) menyediakan API (*Application Programing Interface*) untuk komponen antarmuka pengguna, masukan dan penanganan kesalahan, jaringan(*network*), dan sebagainya. Selain dua komponen diatas, ada satu komponen Java yang tidak bisa ditinggalkan dan merupakan inti dari program Java baik itu J2SE maupun J2EE yaitu *Java Virtual Machine* (JVM), begitu juga dengan J2ME. Tetapi mengingat keterbatasan sumber daya pada perangkat yang digunakan maka pada J2ME ada dua macam *Virtual Machine* yaitu JVM dan KVM (K *Virtual Machine*). JVM digunakan oleh perangkat yang menggunakan CDC, sedangkan KVM digunakan oleh perangkat yang menggunakan CLDC.

Dalam J2ME terdapat beberapa buah profile seperti MIDP (Mobile Information Device Profile), PDAP (*Personal Digital Assistan Profile*), *Fondation Profile*, *Persolal Profile*, serta RMI *Profile*. *Profile* tersebut tersedia untuk kebutuhan-kebutuhan spesifik lainnya sesuai dengan fungsi yang dimiliki oleh suatu perangkat.

Keterhubungan antara configuration dan profile yang ada pada J2ME beserta jenis mesin virtualnya dapat diliat pada gambar 2.5.



Gambar 2.5 Hubungan J2ME dan J2SE

Berikut meruapakan kumpulan paket yang terdapat dalam CLDC 1.0/ MIDP 2.0 antara lain :

1. Paket *java.io*

   Paket ini tergabung dalam CLDC 1.0 dimana paket ini menyediakan sejumlah kelas yang menangani *input/output* data. Dalam aplikasi ini *stream* digunakan sebagai primitif masukan dan keluaran data.

2. Paket *java.lang*

   Palet ini tergabung dalam CLDC 1.0 dimana paket ini menyediakan kelas-kelas fundamental yang diperlukan bahasa pemrograman java, seperti kelas *String*, *Integer*, *System*, dan lain-lain.

3. Paket *java.util*

   Paket ini tergabung dalam CLDC 1.0 diamana paket ini menyediakan kelas-kelas fundamental terutama yang berkaitan dengan *time* dan *date*.

4. Paket javax.microedition.io

   Paket ini tergabung dalam CLDC 1.0 dan MIDP 2.0 diamana paket ini menyediakan kelas-kelas umum yang berkaitan dengan koneksi jaringan.

5. Paket *javax.microedition.midlet*

   Paket ini tergabung dalam MIDP 2.0 dimana paket ini berfungsi untuk mengatur siklus hidup aplikasi dan interaksi antara aplikasi dengan lingkungan tempat aplikasi dijalankan.

6. Paket *javax.microedition.lcdui*

   Paket ini tergabung dalam MIDP 2.0 diamana paket ini menyediakan kelas-kelas yang menangani *user interface* aplikasi.

7. *Paket javax.microedition.rms*

   Paket ini tergabung dalam MIDP 2.0 dimana paket ini menyediakan mekanisme untuk menyimpan data secara permanen untuk kemudian diambil kembali. Data disimpan dalam bentuk *record* yang dapat di-*list* ataupun dienumerasi.

Adapun konfigurasi dan profile yang akan digunakan pada Tugas Akhir ini adalah CLDC versi 1.0 dan MIDP versi 2.0.

II.3.2.3 Paket-Paket Opsional

Paket-paket opsional merupakan paket-paket tambahan yang dibutuhkan oleh aplikasi sehingga pada saat proses *deplovment* paket-paket tersebut perlu

didistribusikan juga sebagai bagian dari aplikasi bersangkutan. Sebagai catatan bahwa paket-paket opsional ini bukan merupakan paket yang dibuat oleh perusahaan alat yang digunakan.

II.3.3 Siklus Hidup Aplikasi J2ME

AMS (*Application Management Software*) merupakan lingkungan tempat sebuah MIDLet dapat di-*install*, dijalankan, diberhentikan maupun di-*uninstall*. AMS akan membuat setiap *instance* baru dari MIDlet dapat mengontrol keadaanya, yaitu dengan cara menjalankan (*start*), mengistirahatkan (*pause*) maupun menghentikannya (*destroy*) secara langsung oleh dirinya sendiri.

Terdapat tiga buah *method* yang harus diimplementasikan oleh setiap MIDlet, yaitu :

1. *Method startApp()*

   AMS akan memanggil *method startApp()* untuk memerintahkan MIDlet agar memperoleh fokus dan menjadikan MIDlet berada dalam keadaan *Active*. Hal ini dapat terjadi ketika MIDlet baru saja dibuat atau MIDlet yang akan kembali diaktifkan dari keadaan *Paused*.

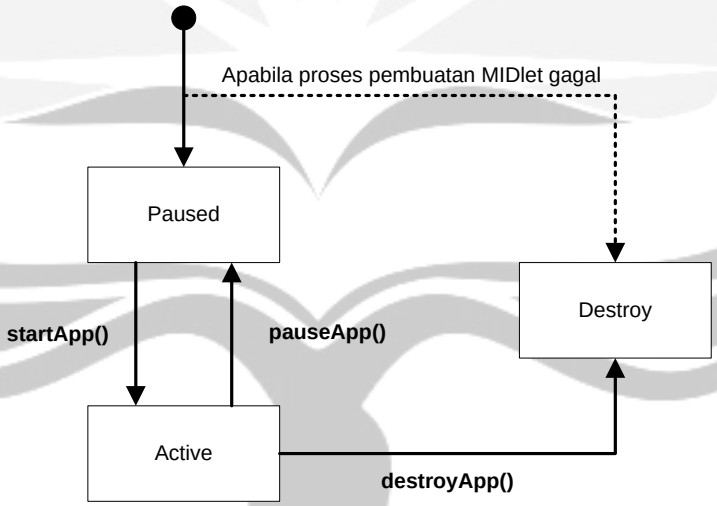
2. *Method pauseApp()*

   AMS memanggil *method pauseApp()* untuk memerintahkan MIDlet agar tidak memiliki fokus dan akan menjadikan MIDlet berada dalam keadaan *Paused*. Dalam keadaan ini, aplikasi tidak dapat memiliki satu pun tampilan UI (*User Interface*). Aplikasi akan kembali berada dalam keadaan *Active* bila diaktivasi ulang.

3. *Method destroyApp()*

   AMS memanggil *method destroyApp()* untuk memerintahkan MIDlet agar membuang atau membebaskan semua *resource* (biasanya berupa file) yang digunakan sekaligus menutup atau menghentikan aplikasi sesegera mungkin. Pemanggilan *method destroyApp()* akan mengakibatkan MIDlet berada dalam keadaaan *Destroyed* sehingga pada saat tersebut MIDlet sudah tidak dapat lagi melakukan pengaksesan terhadap objek *Display*.

Berikut ini merupakan gambar yang kan mengilustrasikan ketiga buah keadaan tersebut dan pada saat kapan MUDlet akan berada dalam keadaan tertentu.



Gambar 2.6 Siklus Hidup MIDlet

Aplikasi-aplikasi yang dibuat didalam telepon seluler dengan menggunakan MIDP disebut dengan MIDlet. MIDlet berupa sebuah kelas abstrak yang merupakan sub kelas dari bentuk dasar aplikasi sehingga antarmuka aplikasi pada Java ME dan aplikasi manajemen pada

perangkat dapat terbentuk. Dalam implementasinya MIDlet memiliki struktur direktori sebagai berikut:

- Src

  Sebagi tempat penyimpanan *source code* untuk MIDlet dan kelas lain yang diperlukan.
- Res

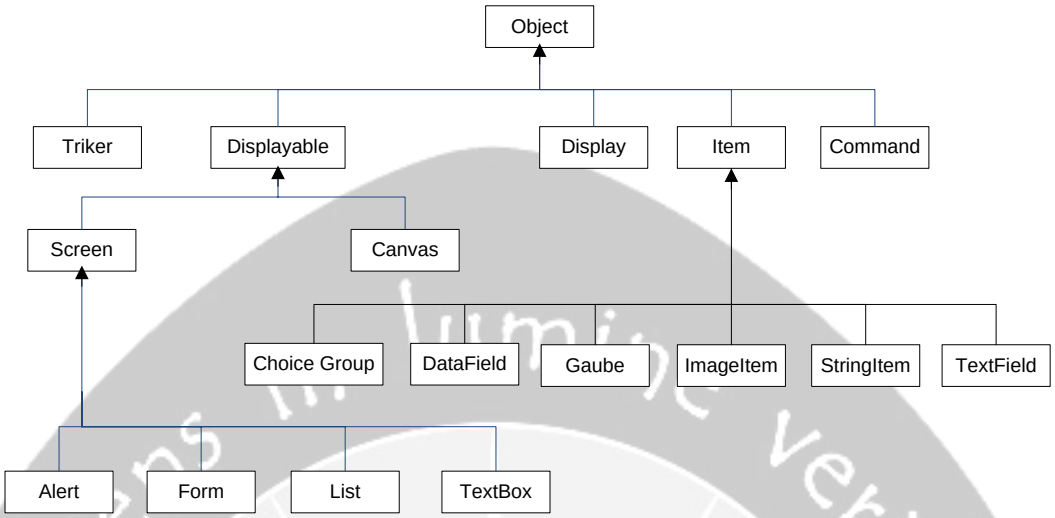
  Sebagai tempat penyimpanan gambar yang diperlukan MIDlet.
- Lib

  Sebagai tempat penyimpanan JAR atau ZIP yang berisi *library* tambahan yang dibutuhkan MIDlet.
- Bin

Sebagi tempat penyimpanan file JAR, JAD, dan file *manifest* yang berisi muatan komponen MIDlet.

II.3.4 Komponen J2ME

J2ME memiliki banyak kelas yang digunakan untuk membuat aplikasi pada *mobile phone*. Kelas-kelas tersebut digunakan untuk membuat tampilan layar, menu, grafik pada *game*, dan sebagainya. Pada Gambar 2.7 ini menunjukan hirarki dari beberapa kelas penting yang banyak digunakan untuk melakukan pembuatan aplikasi GUI dalam MIDP.



Gambar 2.7 Hirarki kelas-kelas penting dalam LCDUI

Berikut ini dijelaskan beberapa kelas yang akan digunakan dalam pengembangan aplikasi Layanan Informasi Isi Ulang Pulsa Super.

1. ***Display***

   Objek *Display* merupakan objek yang bertindak sebagai manajer dari layar tampilan pada aplikasi MIDlet. Pada setiap MIDlet hanya terdapat satu buah objek *Display*. Objek *Display* menyediakan *method* untuk menggambar dan menampilkan elemen user *interface* pada layar.

   Konstraktor dari *Display* adalah :

   Public Static Display getDisplay(MIDlet m)

2. ***Displayable***

   Kelas *Displayable* merupakan kelas abstrak dari semua kelas *user interface*. Kelas *Screen* dan semua kelas turunannya merupakan kelas *user interface* level tinggi sedangkan kelas *Canvas* merupakan kelas *user interface* level rendah. Pada suatu waktu, hanya ada satu objek *Displayable* yang ditampilkan. Objek *Displayable* yang sedang

ditampilksan disebut *current*. Objek *Display* memiliki *method* untuk mengambil objek *Displayable* yang sedang ditampilkan yaitu methot getCUrrent(). Objek *Display* juga memiliki *method* untuk menetapkan objek *Displayable* yang akan ditampilkan yaitu *method* setCurrent(Displayable D).

Konstraktor dari *Displayable* adalah :

Public Displayable getCurrent()

3. ***Screen***

Screen adalah kelas absrak yang merupakan kelas super dari semua kelas *user interface* level tinggi. Kelas turunan dari kelas abstrak *Screen* adalah *Alert*, *Form*, *List*, dan *TextBox*.

4. ***Ticker***

Objek *Ticker* dapat berasosiasi dengan objek sub kelas dari *Screen*. *Ticker* merupakan objek yang berupa tulisan berjalan. Arah dan kecepatan dari *Ticker* tidak dapat diatur secara manual, karena telah diatur oleh sistem dan *Ticker* yang sedang berjalan tidak dapat dihentikan oleh aplikasi.

Konstraktor dari *Ticker* adalah :

Public Ticker(String str)

5. ***Command***

Command adalah objek yang memungkinkan pengguna melakukan aksi. Fungsi objek *Command* sama dengan tombol (*button*) pada aplikasi *desktop* pada komputer. *Command* membutuhkan *interface CommandListener* untuk menangkap kejadian (*event*) dari *Command*. Pada saat membuat aplikasi J2ME pengembangan harus membuat sebuah *Command* untuk

keluar dari aplikasi tersebut karena J2ME tidak mendukung keluar aplikasi secara otomatis.

Konstraktor dari *Command* adalah :

Public Command(String label, int commandType, int priority)

6. ***Alert***

Alert adalah pesan pada layar yang dapat menampilkan teks dan gambar pada pengguna. *Alert* ditujukan untuk menginformasikan pesan kesalahan atau *exception* kepada pengguna. *Alert* juga digunakan untuk menampilkan data dan menunggu selama beberapa waktu tertentu untuk memproses kelas *Displayable* berikutnya.

Konstraktor dari *Alert* adalah :

Public Alert (String title)

Public Alert (String title, String alertText,Image alertImage, AlertType alertType)

7. ***Form***

Form adalah kelas turunan dari kelas abstrak *Screen* yang dapat mengandung elemen-elemen gambar, *text field*, *choice group*, dan elemen lainnya yang merupakan turunan dari kelas *Item*. *Form* memiliki *method append(Item i)* untuk meletakkan sebuah elemen ke dalam *Form*. Parameter *Item* dapat diganti dengan *Image* atau *String* untuk meletakkan elemen gambar atau *string* ke dalam *Form*.

Konstraktor dari *Form* adalah :

Public Form (String title)

Public Form (String title, item[] items)

8. ***List***

List menampilkan himpunan pilihan pada layar. Pengguna dapat memilih diantara pilihan tersebut. Setiap pilihan (elemen dari *list*) mengandung *string* dan gambar. Elemen gambar bersifat opsional, artinya setiap elemen *List* bisa hanya mengandung *string* dan tidak menggandung gambar. *List* dapat memungkinkan pilihan bersifat eksklusif (satu pilihan) maupun *multiple* (lebih dari satu pilihan).

Konstraktor dari *List* adalah :

Public List(String title, int listType)

Public List(String title, int listType, String[] stringElements, Image[] imageElements)

9. ***Textbox***

Textbox adalah sebuah objek yang ditujukan agar pemakai dapat menuliskan teks dan mengeditnya. Jumlah teks yang dapat dimasukan serta tipe teks yang dimasukan dapat ditetapkan dan dibatasi oleh aplikasi.

Konstraktor dari *Textbox* adalah :

Public TextBox(String title,Strng text,int maxSize,int constraints)

10. ***Image***

Sebuah *Image* menyimpan sebuah data grafis gambar. *Image* dapat ditambahkan pada antarmuka level tinggi seperti *Form* dengan menggunakan *method add()*, sedangkan pada antarmuka level rendah seperti *Canvas* dengan menggunakan *method deawImage()*. Antarmuka level tinggi hanya dapat menggunakan gambar yang bersifat *immutable* yaitu

gambar yang tidak dapat dimodifikasi setelah dibuat. Antarmuka level rendah dapat menggunakan gambar yang bersifat *mutable* yang biasanya disimpan dalam memori.

Konstraktor dari *Image* adalah :

Public static Image createImage(String name)

11. ***ChoiceGroup***

ChoiceGroup sangat mirip dengan *List*, kedua kelas tersebut sama-sama mengimplementasikan *interface choice*, yang memiliki banyak *method-method* esensial untuk proses pemilihan *item* dari suatu daftar tertentu. Perbedaannya, objek *ChoiceGroup* tidak dapat berdiri sendiri untuk dijadikan layar aktif melalui method *setCurrent()* yang terdapat pada kelas *Display*, sedangkan *List* bisa.

Konstraktor dari *ChoiceGroup* adalah :

Public ChoiceGroup(String label, int choiceType)

Public ChoiceGroup(String label, int choiceType, String[] stringElements, Image imageElements)

12. ***StringItem***

StringItem mempresentasikan objek teks dan gambar sederhana yang dapat ditempatkan di dalam *form*.

Konstraktor dari *StringItem* adalah :

Public StingItem(String lable, String text)

13. ***TextField***

TextField adalah sebuah objek untuk memasukkan masukan berupa teks ke dalam *form* masukan.

Konstraktor dari *TextField* adalah :

Public TextField(String label, String text, int maxSize, int constraints)

14. *Stream*

Selain kelas yang berhubungan dengan aspek grafis dari aplikasi, juga digunakan kelas untuk komunikasi data, salah satunya adalah *stream*, *Stream* adalah kelas di J2ME yang mendefinisikan masukan dan keluaran data. *Stream* adalah urutan data yang panjangnya tidak diketahui sebelumnya, seperti pada aliran air yang terus mengalir. Stream menerima urutan karakter dari dan atau kepada sebuah proses komunikasi data. Pada J2ME, *stream* menerima ataupun mengirimkan *byte* secara diskrit dimana *byte* dapat merepresentasikan karakter ataupun data lainnya. Beberapa kelas *stream* yang akan digunakan dalam pengembangan aplikasi Layanan Informasi dan Isi Ulang Pulsa super adalah :

1. ByteArryInput/OutputStream
   Berisi tempat penyimpanan *byte* yang dapat menulis atau membaca ke ataupun dari *stream.*
2. DataInput/OutputStream
   Menyediakan aplikasi untuk menulis atau membaca tipe dari sebuah *input* atau *output stream*.
3. PrintStream
   Merupakan ekstensi *OutputStream* dan menyediakan prosedur untuk mencetak atau menampilkan beberapa variasi objek dan nilai data.

II.3.5 Record Management Store (RMS)

Sebuah MIDlet tidak dapat menyimpan data pada suatu *file* karena java tidak diberi *permission* untuk mengakses *resource* telepon seluler secara langsung.

Dengan adanya class javax.microedition.rms maka suatu data dapat disimpan secara *persistent*.

Record Store merupakan class yang menangani kumpulan dari *record* dan seluruh akses ke *record* haruslah melalui perantaraan class *Record Store*. *Class* ini menjamin penyimpanan data yang *atomic* tanpa ada kemungkinan untuk terjadinya data *corrupt*.

Saat sebuah *record* dibuat, *Record Store* akan memberi nomor pada setiap *record* dengan nomor *unique identifier* yang disebut sebagai *record* ID. Penambahan *record* pertama kali ajab duberi ID 1, yang kedua dengan ID2, dan seterusnya. *Record* ID bukanlah suatu index, penghapusan pada sebuah record tidak akan mengakibatkan *renumbering existing record*.

Tiap *Record Store* memiliki nama yang unik untuk membedakannya dengan *Record Store* yang lain. Pada MIDP 1.0, *Record Store* tidaklah dapat untuk digunakan bersama sama antar MIDlet yang berbeda. MIDP 20 sudah mendukung penggunaan *Record Store* yang sama oleh MIDlet yang berbeda.

Jumlah penyimpanan data pada setiap telpon seluler tidaklah sama dan bervariasi. Setiap MIDlet yang menggunakan RMS haruslah menspesifikasikan jumlah minimum dari penyimpanan yang diperlukan pada JAR *mainfest* dan *application descriptor*. Pengaturan nilai minimun yang berlebihan akan dapat mengakibatkan penolakan untuk *install* pada *device* karena *space* yang tersedia tidak sebanyak yang dibutuhkan.

Kecepatan akses pada *presistent* memori lebih lama dari pengaksesan data yang disimpan pada variabel. Pada beberapa *platform*, proses *writing* bisa membutuhkan

waktu yang lama sehingga untuk performance yang lebih bagus digunakan *thread* yang berbeda dari *thread* utama.

Beberapa fungsi dalam kelas *RecordStore:*

a. *public static RecordStore openRecordStore(String RecordStoreName,Boolean createlfMecessary)*
   Fungsi ini digunakan untuk membuka *RecordStore.*
b. *public int addRecord(byte[] data, int offset, int numBytes)*
   Fungsi ini digunakan untuk menambah data pada *RecordStore.*
c. *public void deleteRecord(int RecordlD)*
   Fungsi ini digunakan untuk menghapus data pada *RecordStore.*
d. *public void setRecord(int recordld, byte[] newData, int offset, int numBytes)*
   Fungsi ini digunakan untuk mengganti data pada *RecordStore.*
e. *Public RecordEnumeration emtmerateRecords (RecordFilter filter, RecordComparator comparator, boolean keepUpdated)*
   Fungsi ini digunakan untuk membangun sebuah enumerasi untuk mengambil *record* dari *record store.*
f. *public int getNextRecordID()*
   Fungsi ini digunakan untuk membaca data selanjutnya. pada *RecordSlore.*
g. *public byte[] getRecord(int recorded)*
   Fungsi ini digunakan untuk mendapatkan data pada. *RecordStore*
h. *public void closeRecordStore()*
   Fungsi ini digunakan untuk menutup *RecordStore.*

i. *public static void deleteRecordStore(String RecordStoreName)*

   Fungsi ini digunakan untuk menghapus *RecordStore.*

II.4 HTTP Connection

Pada MIDP 1.0, satu satunya protokol yang dapat diimplementasikan adalah Protokol HTTP. Melalui *class HttpConnection handphone* dapat berkomunikasi dengan *web server.* HTTP dikenal sebagai protokol *request / response* dimana *client* memulai komunikasi dengan mengirimkan *request* kepada suatu alamat *server* atau URL dan *server* memberi balasan berupa *response.*

Dasar Penggunaan HTTP connection connection dari *handphone* adalah sebagai berikut :

1. Untuk membuka suatu koneksi dilakukan dengan memanggil fungsi Connection.open().
2. *Set request method*, tahap ini bisa diabaikan.
3. *Set header* dari *request* yang diperlukan, tahap ini juga bisa diabaikan.
4. Panggil fungsi openOutputStream() atau openDataOutputStream() untuk membuka *stream* yang digunakan untuk menulis data yang akan dikirim.
5. Tulislah data yang akan dikirim (jika ada) kedalam *output stream* lalu *flush stream.*
6. Panggil fungsi openInputStream() atau openDataInputStream() untuk mendapatkan *response* berupa *input stream.*
7. Dapatkan *response code* dengan memanggil getResponseCode(), jika *request*nya berhasil, baca data dari *input stream.*

8. Lalu tutup *connection,input stream*, dan *output stream.*

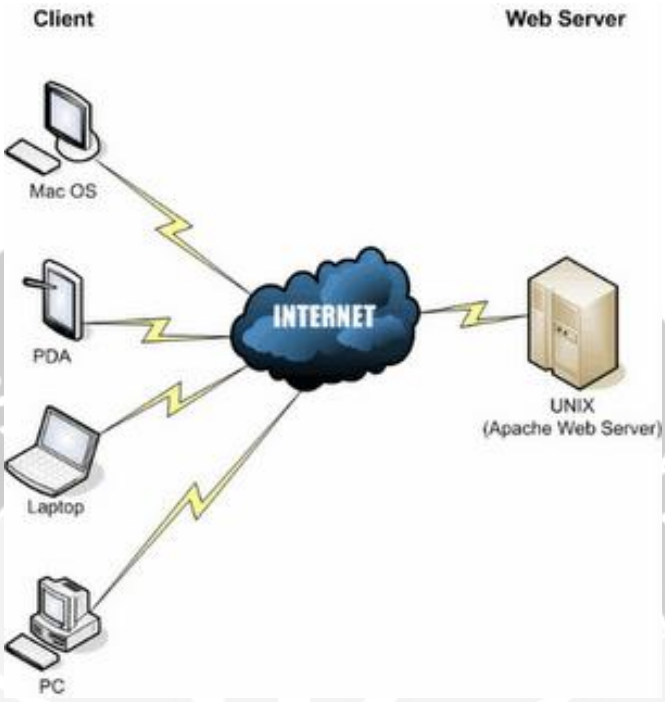
II.5 Web Server Apache

Pada saat ini web server merupakan salah satu server penting yang digunakan selain selain e-mail server, FTP, dan *news server.* Web server dirasasangat penting karena dengan adanya web server maka informasi yang ada di Intemet dapat ditampilkan dengan lebih efektif, dinamis, dan menarik. Karena keunggulan ini membuat web dapat diterima di semua kalangan. Pada saat ini hampir semua universitas dan perasahaan komersial telah memiliki sebuah atau beberapa web server.

WWW server atau lebih sering disebut dengan Web server adalah server Internet yang melayani koneksi transfer data dalam protokol HTTP. Web server telah didesain untuk dapat melayani bermacam-macam jenis data, mulai dari teks, gambar, sampai dengan suara. Pada umumnya Web server melayani data dalam bentuk HTML. Web server juga dapat juga dikombinasikan dengan *wireless* Intemet. Dengan menggabungkan web server dengan sebuah WAP *gateway,* maka web server dapat menjadi WAP server, yang siap melayani akses *wireless* Intemet pada ponsel yang telah memiliki fitur WAP atau GPRS.

Apache adalah web server yang paling banyak digunakan. Hal Ini disebabkan oleh beberapa faktor, yaitu:

- Kecepatan
- Performa
- Harganya gratis



Gambar 2.8 Web Server

II.6 PHP

II.6.1 Sekilas Tentang PHP

PHP yang merupakan singkatan dari PHP *Hypertext Prepocessor* adalah suatu bahasa yang bersifat *server side* yang didesain khusus untuk aplikasi web. PHP dapat disisipkan diantara bahasa HTML. Karena bahasa *server side*, maka bahasa PHP akan dieksekusi di server, sehingga yang dikirimkan ke *browser* adalah "hasil jadi" dalam bentuk HTML, dan kode PHP tidak terlihat lagi (Kadir,2001).

II.6.2 Kelebihan PHP

PHP mempunyai kelebihan-kelebihan, yaitu :

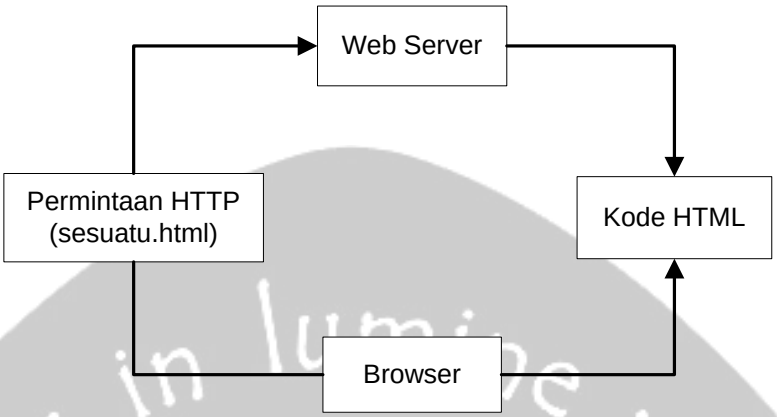
- PHP mudah dibuat dan kecepatan akses tinggi PHP dapat berjalan dalam *web server* yang berbeda dan

dalam sistem operasi yang berbeda pula. PHP dapat berjalan di sistem operasi UNIX, Windows98,Windows NT, dan Macintosh.

- PHP diterbitkan secara gratis.
- PHP juga dapat berjalan pada *web server* Microsoft Personal Web Server, Apache, IIS, Xitami, dan sebagainya.
- PHP adalah termasuk bahasa yang *embedded* (bisa ditempel atau diletakkan dalam tag HTML).
- PHP termasuk *server-side programming.*

II.6.3 Cara Kerja PHP

Model kerja HTML diawali dengan permintaan suatu halaman web oleh *browser.* Berdasarkan URL (*Uniform Resource Locator*) atau dikenal dengan sebutan alamat Internet, *browser* mendapatkan alamat dari *web server*, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh *web server*. Informasi yang disampaikan ke *web server* antara lain adalah nama *browser*, versinya, dan sistem operasinya. Selanjutnya, *web server* akan mencarikan berkas yang diminta dan memberikan isinya ke *browser*. *Browser* yang mendapatkan isinya segera melakukan proses penerjemahan kode HTML dan menampilkannya ke layar pemakai seperti Gambar 2.9



Gambar 2.9 Skema HTML

\

jika yang diminta adalah sebuah halaman PHP, maka prinsipnya serupa dengan kode HTML. Hanya saja, ketika berkas PHP yang diminta didapatkan oleh *web server*, isinya segera dikirimkan ke mesin PHP dan mesin inilah yang memproses dan memberikan hasilnya (berupa kode HTML) ke *web server.* Selanjutnya, *web server* menyampaikan ke klien.

II.7 MySQL

II.7.1 Sekilas Tentang MySQL

MySQL adalah *Relational Database Management System* (RDMS) yang didistribusikan secara gratis dibawah lisensi *General Public License* (GPL). Setiap orang bebas menggunakannya, tetapi tidak boleh dijadikan produk turunan yang bersifat komersial (Sutarman, 2003). MySQL merupakan turunan dari salah satu konsep utama dalam *database* yaitu *Structured Query Language* (SQL). SQL adalah sebuah konsep pengoperasian *database* terutama untuk pemilihan dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

MySQL juga merupakan sebuah database server yang handal dan aplikasi ini memiliki versi unix dan windows. Apabila membuat aplikasi yang membutuhkan interface ke server MySQL telah disediakan library dari MySQL. MySQL merupakan sebuah aplikasi Relational Database Management Server RDBMS yang sangat cepat dan kokoh. MySQL merupakan sebuah server basis data(database server) yang banyak digunakan di internet karena keandalannya dan juga karena sifatnya yang shareware. Layaknya database yang lain MySQL mengenal bahasa SQL. SQL (Structured Query Language) adalah bahasa standar yang digunakan untuk mengakses server basis data. Bahasa ini pada awalnya dikembangkan oleh IBM, namun telah diadopsi dan digunakan sebagai standar industri.

Dalam konteks bahasa SQL, pada umumnya informasi tersimpan dalam tabel-tabel yang secara logik merupakan struktur dua dimensi yang terdiri atas baris-baris data (row atau record) yang berada dalam satu atau lebih kolom (column). Baris pada tabel sering disebut sebagai attributes atau field. Keseluruhan tabel itu dihimpun dalam satu kesatuan yang disebut basis data. Ketika membandingkan MySQL dengan sistem basis data yang lain, pikirkan apa yang paling penting, 'performance' (dayaguna), 'support', keistimewaan, kebebasan dan pembatasan dalam penggunaan dan harga adalah faktor-faktor yang perlu dipikirkan. Sebagai *database server,* MySQL termasuk unggul dibandingkan *database server* lainnya dalam query data. Hal ini dapat dibuktikan melalui kecepatan MySQL yang bisa sepuluh kali lebih

cepat dari PostgreSQL dan lima kali lebih cepat dibanding Interbase.

II.7.2 Kelebihan MySQL

MySQL merupakan *database server* yang memiliki konsep *database modern*. MySQL mempunyai beberapa kelebihan, antara lain :

- *Portability*
  MySQL dapat berjalan stabil pada berbagai sistem operasi di antaranya adalah Windows, Linux, FreeBSD, Mac OS X Server, Solaris, Amiga, HP-UX, dan lain-lain.
- *Open Source*
  MySQL didistribusikan secara gratis (*open source)*, dibawah lisensi GPL sehingga dapat digunakan tanpa dipungut biaya sepeser pun.
- *Multiuser*
  MySQL dapat digunakan oleh beberapa *user* dalam waktu yang bersamaan tanpa mengalami masalah atau konflik. Hal ini memungkinkan sebuah *daatabase server* MySQL dapat diakses *client* secara bersamaan.
- *Performance Tuning*
  MySQL memiliki kecepatan yang baik dalam menangani *query* sederhana. MySQL dapat memproses lebih banyak SQL per satuan waktu.
- *Column Types*
  MySQL memiliki tipe kolom yang sangat kompleks, seperti *signed/unsigned integer, float, double, char, varchar, text, blob, date, time, datetime, timestamp, year, set* dan *enum.*

- *Command* dan *Functions*

  MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah SELECT dan WHERE dalam query.
- *Security*

  MySQL memiliki beberapa lapisan *security* seperti level subnetmask, nama host, dan izin akses *user* dengan sistem yang mendetail serta *password* yang menggunakan sistem enkripsi.
- *Scalability* dan *Limits*

  MySQL mampu menangani database dalam skala besar, dengan jumlah *records* lebih dari 50 juta dan 60 ribu tabel serta 5 milliar baris. Batas index yang dapat ditampung oleh MySQL adalah sebanyak 32 index dari tiap tabel.
- *Connectivity*

  MySQL dapat melakukan koneksi dengan *client* melalui penggunaan protokol TCP/IP, Unix soket (Unix), atau Namd Pipes (NT).
- *Localication*

  MySQL dapat mendeteksi pesan kesalahan (*error code*) pada *client* dengan menggunakan lebih dari dua puluh bahasa.
- *Interface*

  MySQL memiliki *interface* terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi *Application Programming Interface* (API).
- *Table Structure*

  MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABEL, dibandingkan dengan *database* lainnya seperti PostgreSQL atau Oracle.

- *Clients* dan *Tools*

  MySQL dilengkapi dengan berbagai tool yang dapat digunakan untuk administrasi *database.* Pada setiap tool dalam MySQL disertakan petunjuk *online.*