BAB II

LANDASAN TEORI

2.1 Aplikasi Mobile

Aplikasi *mobile* saat ini sangat dibutuhkan karena alat-alat telekomunikasi yang tersebar di seluruh dunia membutuhkan aplikasi-aplikasi yang dapat memepermudah pekerjaan penggunanya dimanapun dan kapanpun terutama dalam hal informasi. Aplikasi ini dapat diakses melalui perangkat nirkabel seperti pager, seperti telepon seluler dan PDA.

2.1.1 Pengertian Aplikasi

Menurut Buyens (2001) aplikasi adalah satu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas. Misalnya termasuk perangkat lunak perusahaan , software akuntansi , perkantoran , grafis perangkat lunak dan pemutar media. Dapat disimpulkan bahwa aplikasi merupakan *software* yang berfungsi untuk melakukan berbagai bentuk pekerjaan atau tugas-tugas tertentu seperti penerapan, penggunaan dan penambahan data.

Program aplikasi merupakan program siap pakai. Program yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain. Contoh-contoh aplikasiialah program pemproses kata dan Web Browser. Aplikasi akan menggunakan sistem operasi (OS) komputer dan aplikasi yang lainnya yang mendukung.

Klasifikasi aplikasi dapat dibagi menjadi 2 (dua) yaitu:

1. Aplikasi software spesialis, program dengan dokumentasi tergabung yang dirancang untuk menjalankan tugas tertentu.
2. Aplikasi paket, dengan dokumentasi tergabung yang dirancang untuk jenis masalah tertentu.

2.1.2 Pengertian Mobile

Mobile menurut (http://agusbarupunyablog.blogspot.com/2010/10/pengertian-aplikasi-mobile.html, 2012), diartikan sebagai perpindahan yang mudah dari satu tempat ke tempat yang lain, misalnya telepon mobile berarti bahwa terminal telepon yang dapat berpindah dengan mudah dari satu tempat ke tempat lain tanpa terjadi pemutusan atau terputusnya komunikasi

2.1.2.1 Karakteristik Perangkat Mobile

Perangkat mobile memiliki banyak jenis dalam hal ukuran, desain dan layout, tetapi mereka memiliki kesamaan karakteristik yang sangat berbeda dari sistem desktop, yaitu antara lain :

a) Ukuran yang kecil

   Perangkat mobile memiliki ukuran yang kecil. Konsumen menginginkan perangkat yang terkecil untuk kenyamanan dan mobilitas mereka.

b) Memory yang terbatas

   Perangkat mobile juga memiliki memory yang kecil, yaitu primary (RAM) dan secondary (disk). Pembatasan ini adalah salah satu faktor yang

mempengaruhi penulisan program untuk berbagai jenis dari perangkat ini. Dengan pembatasan jumlah dari memory, pertimbangan-pertimbangan khusus harus diambil untuk memelihara pemakaian dari sumber daya yang mahal ini.

c) Daya proses yang terbatas

Sistem mobile tidaklah setangguh rekan mereka yaitu desktop. Ukuran, teknologi dan biaya adalah beberapa faktor yang mempengaruhi status dari sumber daya ini. Sepertiharddisk dan RAM, Anda dapat menemukan mereka dalam ukuran yang pas dengan sebuah kemasan kecil.

d) Mengkonsumsi daya yang rendah

Perangkat mobile menghabiskan sedikit daya dibandingkan dengan mesin desktop.

Perangkat ini harus menghemat daya karena mereka berjalan pada keadaan dimana daya yang disediakan dibatasi oleh baterai-baterai.

e) Kuat dan dapat diandalkan

Karena perangkat mobile selalu dibawa kemana saja, mereka harus cukup kuat untuk menghadapi benturan-benturan, gerakan, dan sesekali tetesan-tetesan air.

f) Konektivitas yang terbatas

Perangkat mobile memiliki bandwith rendah, beberapa dari mereka bahkan tidak tersambung. Kebanyakan dari mereka menggunakan koneksi wireless.

g) Masa hidup yang pendek

Perangkat-perangkat konsumen ini menyala dalam hitungan detik kebanyakan dari mereka selalu menyala. Coba ambil kasus sebuah handphone, mereka booting dalam hitungan detik dan kebanyakan orang tidak mematikan handphone mereka bahkan ketika malam hari. PDA akan menyala jika anda menekan tombol power mereka.

2.1.3 Pengertian Aplikasi Mobile

Menurut Buyens (2001) aplikasi *mobile* berasal dari kata *application* dan *mobile*. *Application* yang artinya penerapan, lamaran, penggunaan. Secara istilah aplikasi adalah program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju sedangkan *mobile* dapat di artikan sebagai perpindahan dari suatu tempat ke tempat yang lain.

Kata *mobile* mempunyai arti bergerak atau berpindah, sehingga aplikasi mobile menurut Rangsang Purnama (2010) adalah sebutan untuk aplikasi yang berjalan di *mobile device* . Dengan menggunakan aplikasi mobile, dapat dengan mudah melakukan berbagaii macam aktifitas mulai dari hiburan, berjualan, belajar, mengerjakan pekerjaan kantor, browsing dan lain sebagainya.

Pemanfaatan aplikasi mobile untuk hiburan paling banyak digemai oleh pengguna telepon seluler, karena dengan memanfaatkan adanya fitur game, music player, sampai video player membuat kita mejadi semakin mudah menikmati hiburan kapan saja dan dimanapun.

Perangkat mobile memiliki banyak jenis dalam hal ukuran,desai layout, tetapi memreka memiliki kesamaan karakteristik yang sanagt berbeda dari dekstop system. Perangkat mobile memiliki memory yang kecil.

2.2 Sejarah Singkat Kota Bandung

Menurut (http://id.wikipedia.org/wiki/Kota_Bandung 7 mei 2012) Kata "Bandung" berasal dari kata *bendung* atau *bendungan* karena terbendungnya sungai Citarum oleh lava Gunung Tangkuban Perahu yang lalu membentuk telaga. Legenda yang diceritakan oleh orang-orang tua di Bandung mengatakan bahwa nama "Bandung" diambil dari sebuah kendaraan air yang terdiri dari dua perahu yang diikat berdampingan yang disebut *perahu bandung* yang digunakan oleh Bupati Bandung, R.A. Wiranatakusumah II, untuk melayari Ci Tarum dalam mencari tempat kedudukan kabupaten yang baru untuk menggantikan ibukota yang lama di Dayeuhkolot.

Kota Bandung secara geografis memang terlihat dikelilingi oleh pegunungan, dan ini menunjukkan bahwa pada masa lalu kota Bandung memang merupakan sebuah telaga atau danau. Legenda Sangkuriang merupakan legenda yang menceritakan bagaimana terbentuknya danau Bandung, dan bagaimana terbentuknya Gunung Tangkuban Perahu, lalu bagaimana pula keringnya danau Bandung sehingga meninggalkan cekungan seperti sekarang ini. Air dari danau Bandung menurut legenda tersebut kering karena mengalir melalui sebuah gua yang bernamaSangkyang Tikoro.

Daerah terakhir sisa-sisa danau Bandung yang menjadi kering adalah Situ Aksan, yang pada tahun 1970-an masih merupakan danau tempat berpariwisata, tetapi saat ini sudah menjadi daerah perumahan untuk pemukiman.

Kota Bandung mulai dijadikan sebagai kawasan pemukiman sejak pemerintahan kolonial Hindia-Belanda, melalui Gubernur Jenderalnya waktu itu Herman Willem Daendels, mengeluarkan surat keputusan tanggal 25 September 1810 tentang pembangunan sarana dan prasarana untuk kawasan ini. Dikemudian hari peristiwa ini diabadikan sebagai hari jadi kota Bandung.

Kota Bandung secara resmi mendapat status *gemeente* (kota) dari Gubernur Jenderal J.B. van Heutsz pada tanggal 1 April 1906[11] dengan luas wilayah waktu itu sekitar 900 ha, dan bertambah menjadi 8.000 ha di tahun 1949, sampai terakhir bertambah menjadi luas wilayah saat ini.[12]

2.3 Pemandu Pariwisata

Pemandu pariwisata atau dapat disebut juga *tour guide* / pramuwisata diartikan sebagai seseorang yang memberi penerangan, penjelasan, serta petunjuk kepada wisatawan (*tourist*) dan *travelers* lainnya, tentang segala sesuatu yang hendak di lihat, disaksikan oleh wisatawan dan *travellers* yang bersangkutan, bilamana mereka berkunjung pada suatu objek, tempat atau daerah tertentu.

2.3.1 Berdasarkan Keahlianya

Berdasarakan keahliannya pramuwisata dibedakan menjadi beberapa jenis diantaranya yaitu :

1. Pramuwisata umum (General guide)

   Pramuwisata yang mempunyai pengetahuan mengenai kebudayaan, kekayaan alam dan aspirasi kehidupan bangsa atau penduduk sesuatu wilayah secara umum yang mempunyai ijin untuk memberikan bimbingan perjalanan dan penerangan kepariwisataan dengan mempergunakan satu atau beberapa bahasa tertentu terhadap wisatawan baik sebagai perseorangan atau secara berkelompok.

2. Pramuwisata khusus (Special guide)

   Pramuwisata yang mempunyai pengetahuan mengenai objek wisata secara khusus mendalam mengenai satu atau lebih objek wisata seperti kebudayaan, arkeologi, sejarah teknik perdanganan, keagamaan, ilmiah margasatwa, perburuan, dan lain selanjutnya yang mempunyai ijin usaha atau kegiatan untuk membimbing perjalanan dan memberikan penerangan kepada wisatawan baik sebagai perseorangan atau sebagai kelompok dengan menggunakan satu atau beberapa bahasa tertentu.

3. Tour Conductor (Pembimbing darmawisata)

   Pramuwisata senior, yang mempunyai kartu tanda pramuwisata untuk memimpin perjalanan satu kelompok wisatawan yang melakukan perjalanan di suatu wilayah atau beberapa Negara guna memberikan asistensi perjalanan, bimbingan dan penerangan mengenai objek wisata, kebudayaan, kekayaan

alam dan aspirasi kehidupan dari penduduk / bangsa wilayah yang dijelajahnya.

4. Pramuwisata Pengemudi (Guide driver)

Pramuwisata yang mempunyai kartu tanda pramuwisata untuk memberikan bimbingan perjalanan dan penerangan umum mengenai objek wisata, kebudayaan, kekayaan alam dan aspirasi kehidupan bangsa di samping kedudukannya sebagai pengemudi kendaraan umum seperti taxi, touring coach, bus dan sebagainya.

2.3.2 Berdasarkan Hubungan Kerja

Dalam hubungan kerja pramuwisata dibedakan sebagai berikut antara lain:

1. Pramuwisata tour operator (*tour operator guide*)

   Pramuwisata yang bertugas mengatur, memimpin dan menyelenggarakan perjalanan wisata termasuk perlengkapan perjalanan, baik di dalam maupun ke luar negri atas nama perusahaan perjalanan tempat ia berkerja.

2. Pramuwisata objek wisata (*tourist object guide)*

   Pramuwisata yang memberikan penerangan, menunjukkan dan bimbingan ke suatu atau lebih objek wisata.

3. Pramuwisata bebas (*Independent guide*)

   Pramuwisata yang tidak terikat dalam kontrak kerja dengan suatu perusahaan biro perjalanan tetapi hanya di besarkan untuk beroperasi melakukan kegiatan pembimbingan wisatawan dan memberikan penerangan kepada wisatawan bilamana ia disewa atau bertindak untuk

salah satu tour operator atau travel agent dan mempunyai ijin usaha dan kegiatan untuk memangku tugas pramuwisata.

2.3.3 Pengertian Pariwisata

Menurut Yoeti (1996) dalam I Gusti Putu Sutarma pariwisata adalah suatu perjalanan yang dilakukan untuk sementara waktu yang diselenggarakan dari suatu tempat ke temapt lain dengan maksud bukan untuk berusaha (business) atau untuk mencari nafkah di tempat yang dikujungi, tetapi semata – mata untuk menikmati perjalanan tersebut guna pertamasyaan dan rekreasi atau untuk memenuhi keinginan yang beraneka ragam.

Pariwisata menurut I Nyoman Mandia (2008) dalah salah satu jenis industri yang baru yang mampu mempercepat pertumbuhan ekonomi dan penyedian lapangan kerja, peningkatan penghasilan, standar hidup serta menstimulasi sektor – sektor produktif lainnya.

Pariwisata atau turisme menurut wikipedia adalah suatu perjalanan yang dilakukan untuk rekreasi atau liburan, dan juga persiapan yang dilakukan untuk aktivitas ini. Seorang wisatawan atau turis adalah seseorang yang melakukan perjalanan paling tidak sejauh 80 km (50 mil) dari rumahnya dengan tujuan rekreasi.

Menurut Mathies on dan Wall (1982) dalam Prof.Dr.I Gde Pitana,M.Sc. dan I Ketut Surya Diarta,SP.,MA mengatakan bahwa pariwisata mencakup tiga elemen utama, yaitu :

1. *A dynamic elemen*, yaitu travel ke suatu destinasi wisata
2. *A static elemen,* yaitu singgah di daerah tujuan
3. *A consequential elemen*, atau akibat dari dua hal di atas (khususnyat terhadap masyarakat lokal), yang meliputi dampak ekonomi, sosila dan fisik dari adanya kontak dengan wisatawan.

2.3.4 Tujuan Pariwisata

Adapun tujuan pengembangan pariwisata Indonesia antara lain :

1. Meningkatkan pendapatan devisa pada khususnya dan pendapatan Negara dan masyarakat pada umunya, perluasan kesempatan serta lapangan kerja dan mendorong kegiatan – kegiatan industri penunjang dan industri – industri sampigan lainya.
2. Memperkenalkan dan mendayagunakan keindahan alam dan kebudayaan Indonesia.
3. Meningkatkan persaudaraan/persahabatan nasional dan internasional.

2.4 JAVA

Java menurut Shalahuddin dan Rosa (2010) adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *standalone* ataupun pada lingkungan jaringan. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis

model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (*bytecode*) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda, java dikenal pula dengan slogannya, "*Tulis sekali, jalankan di mana pun*". Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

2.4.1 Edisi Java

Menurut M. Shalahuddin (2008) Java adalah bahasa yang dapat di jalankan dimanapun dan disembarang platform apapun, diberagam lingkungan : internet, intranet, consumer Electronic products dan computer Applications. The Java 2 platform tersedia dalam 3 edisi untuk keperluan berbeda. Untuk beragam aplikasi yang dibuat dengan bahasa java, java dipaketkan dalam edisi – edisi berikut :

1. Java 2 Standard Edition (J2SE)

Menurut M. Shalahuddin (2008) The Java 2 Standard Edition menyediakan lingkungan pengembangan yang kaya akan fitur, stabil, aman dan cross – platform. Edisi ini mendukung konektivitas basis data, rancangan antar

muka pemakai, masukan / keluaran dan pemograman jaringan dan termasuk sebagai paket – paket dasar bahasa Java.

2. Java 2 Enterprise Edition (J2EE)

Menurut M. Shalahuddin (2008)The Java Enterprise Edition menyediakan *tool* untuk membangun dan menjalankan *multitier enterprise applications*. J2EE bersisi paket – paket di J2SE ditambah paket – paket untuk mendukung pengembangan Enterprise JavaBeeans, Java Servlets, Java Server Pages, XML dan kendali transaksi yang fleksibel

3. Java 2 Micro Edition (J2ME)

Menurut M. Shalahuddin (2008) The Java 2 Micro Edition selain menyedikan bahasa Java yang sama, unggul dalam portabilitas (kemampuan dapat dijalankan dimanapun), *safe network delivery*, seperti J2SE dan J2EE. Aplikasi-aplikasi dapat diskalakan (dimampukan) agar dapat bekerja dengan J2SE dan J2EE.

J2ME untuk beragam consumer electronic product seperti pager, smart card, cell phone, handled PDA, dan set top box.

2.4.2 Gambaran J2ME (Java 2 Micro Edition)

Menurut M. Shalahuddin (2008) The Java 2 Micro Edition selain menyedikan bahasa Java yang sama, unggul dalam portabilitas (kemampuan dapat dijalankan dimanapun), *safe network delivery*, seperti J2SE dan J2EE. Aplikasi-aplikasi dapat diskalakan (dimampukan) agar dapat bekerja dengan J2SE dan J2EE.

Program *Java ME*, seperti semua program *Java* adalah diterjemahkan oleh *Virtual Machine (VM)*. Program-program tersebut dikompile ke dalam *bytecode* dan diterjemahkan oleh *Java Virtual Machine (JVM)*. Ini berarti bahwa program-program tersebut tidak berhubungan langsung dengan perangkat keras. *Java ME* menyediakan suatu *interface* yang sesuai dengan perangkat keras. Aplikasi-aplikasi tersebut cukup dikompile sekali dan mampu dijalankan pada mesin yang berbeda. Inti dari *Java ME* terletak pada *configuration* dan *profileprofile*.

Suatu *configuration* menggambarkan lingkungan *runtime* dasar dari suatu sistem *Java ME*. Ia menggambarkan *core library*, *virtual machine*, fitur keamanan dan jaringan.



Gambar 2.1 *Arsitektur Java ME*
Sumber : M.Shalahudin dan Rosa A.S (2010:6)

Sebuah *profile* memberikan *library* tambahan untuk suatu kelas tertentu pada sebuah perangkat. *profile-profile* menyediakan *user interface (UI) API, persistence, messaging library*, dan sebagainya. Satu *set library* tambahan atau *package* tambahan menyediakan kemampuan program tambahan. Pemasukan *package* ini ke dalam perangkat *Java ME* dapat berubah-ubah karena

tergantung pada kemampuan sebuah perangkat.

Sebagai contoh, beberapa perangkat *MIDP* tidak memiliki *Bluetooth builtin*, sehingga *Bluetooth API* tidak disediakan dalam perangkat ini.

2.4.3 Konfigurasion

Suatu *configuration* menggambarkan fitur minimal dari lingkungan lengkap *Java runtime*. Untuk menjamin kemampuan portabilitas dan interoperabilitas yang optimal diantara berbagai macam perangkat yang dibatasi sumber dayanya (memory, prosesor, koneksi yang dibatasi), *configuration* tidak menggambarkan fitur tambahan. Suatu *configuration Java ME* menggambarkan suatu komplemen yang minimum dari teknologi *Java*. Adalah merupakan tugas *profile-profile* untuk menggambarkan tambahan *library* untuk suatu kategori perangkat tertentu. *Configuration* menggambarkan:

a) Subset bahasa pemrograman *Java*

b) Kemampuan *Java Virtual Machine (JVM)*

c) *Core platform libraries*

d) Fitur sekuriti dan jaringan.

J2ME terdiri dari dua configuration yaitu :

1) Connected Limited Device Configuration (CLDC) Digunakan pada device yang terbatas dalam hal memori yang kecil dan prosesor yang tidak begitu cepat. Hanya terdiri dari sebagian kecil class librari J2SE seperti java.lang dan java.util, kemudian ditambahkan paket untuk aplikasi mobile javax.microedition.io.CLDC banyak digunakan pada handphone dan PDA (Personal Digital Asistants).

2) Connected Device Configuration (CDC) Berisikan semua dari java virtual machine (JVM), dan ditambahkan paket untuk aplikasi mobile sehingga membutuhkan memori yang lebih besar dan prosesor yang lebih cepat dari CLDC.CDC banyak digunakan pada internet tv, car tv dan nokia communicator .

2.4.4 Profile

Suatu *profile* menggambarkan set-set tambahan dari *API* dan fitur untuk pasar tertentu, kategori perangkat atau industri. Sementara *configuration* menggambarkan *library* dasar, *profile-profile* menggambarkan *library* yang penting untuk membuat aplikasiaplikasi efektif. *Library* ini memasukkan *user interface*, jaringan dan penyimpanan API.

Profile merupakan class librari tambahan pada configuration yang digunakan untuk pengelompokan spesifikasi tertentu (domainspesific), dan digunakan untuk mendukung paket-paket opsional agar dapat mengakses virtual machine. Profile merupakan kebalikan dari configuration yaitu mengatur hal-hal yang spesifik untuk sebuah device. Terdapat beberapa profile, namun profile pertama yang selesai dirilis adalah Mobile Information Device Profile (MIDP) yang bertugas untuk menjalankan aplikasi pada handphone atau pada pager. Satu device dapat terdiri dari beberapa profile seperti pada PDA, terdapat MIDP, dan Personal Basis Profile (PBP) yang digunakan untuk membuat tampilan grafik seperti window.

2.4.5 Optional Packages

Optional package adalah kumpulan API yang berisikan fungsi-fungsi tertentu yang tidak terdapat pada configuration dan profile. Optional package adalah kumpulan API yang berisikan fungsi-fungsi tertentu yang tidak terdapat pada configuration dan profile.

2.4.6 MIDP

The Mobile Information Device Profile (MIDP) berada di atas dari *CLDC*. Kita tidak bisa menulis aplikasi mobile hanya dengan menggunakan *CLDC API*. Kita harus tetap memanfaatkan *MIDP* yang mendefinisikan *UI*. Spesifikasi *MIDP*, kebanyakan seperti *CLDC* dan *API* lainnya sudah digambarkan melalui *Java Community Process (JCP)*. *JCP* melibatkan sebuah kelompok ahli berasal dari lebih dari 50 perusahaan, yang terdiri atas pabrik perangkat mobile, pengembang software. *MIDP* terus berkembang, dengan versi-versi masa depan yang telah lulus dari proses ketat *JCP*.

Spesifikasi *MIDP* menggambarkan suatu perangkat *MID* yang memiliki karakteristik-karateristik ini sebagai batas minimum:

1) Tampilan:
    a) Ukuran Layar: 96x54
    b) kedalaman tampilan: 1-bit
    c) Ketajaman pixel: sekitar 1:1
2) Masukan:

a) Satu atau lebih mekanisme user-input: satu keybboard, dua keyboard, atau touch screen

3) Memory:
   a) 256 kilobytes of non-volatile memory untuk implementasi MIDP.
   b) 8 kilobytes of non-volatile memory for applicationcreated persistent data
   c) 128 kilobytes of volatile memory for the Java runtime (e.g., the Java heap)
4) Jaringan:
   a) dua jalur, wireless, bandwidth terbatas
5) Sound:
   a) Kemampuan untuk memainkan nada-nada

2.4.7 MIDlet

Suatu aplikasi *MIDP* disebut *MIDlet*. Perangkat *application management software (AMS)* berinteraksi langsung dengan *MIDlet* dengan *method MIDlet create, start, pause, dan destroy*. *MIDlet* adalah bagian dari *package javax.microedition.midlet*. Sebuah *MIDlet* harus di-*extend* dari *class MIDlet*. Dan dapat meminta parameter dari *AMS* seperti dirumuskan dalam *java application descriptor (JAD)*. Suatu *MIDlet* tidak harus memiliki (dan memang harus tidak mempunyai) sebuah *method public static void main(String[] argv)*. *Method* tersebut tidak akan dikenal lagi oleh *AMS* sebagai titik awal sebuah program.

2.4.8 Library Pendukung

Aplikasi ini memerlukan beberapa software pendukung untuk menambah kebutuhan sistem. Software pendukung itu antara lain :

a) AvetanaOBEX

AvetanaOBEX adalah salah satu paket pendukung atau paket tambahan yang ada pada bahasa pemrograman java, yang berfungsi untuk membantu dalam pengiriman file. J2SE secara default tidak mendukung JSR82, paket tambahan untuk JSR82 khusus untuk OBEX pada J2SE salah satunya adalah AvetanaOBEX yang dapat didownload pada http://avetana.sourceforge.net

b) Bluecove

Bluecove merupakah paket bluetooth API yang digunakan untuk operating sistem window, bluecove mengimplementasikan semua JSR82 tanpa OBEX. Bluecove dapat didownload pada http://bluecove.sourceforge.net.

c) COMM API

COMM API juga merupakan paket tambahan pada program java yang berfungsi untuk mengakses serial port yang ada pada komputer. J2SE secara default tidak dapat melakukan akses pada serial port seperti USB atau COM port. Sedangkan device bluetooth yang digunakan adalah USB bluetooth yang terinstal pada operating sistem, sehingga dibutuhkan COMM API untuk melakuskan kontrol hardware.

2.5 Pengertian UML

UML menurut Yasmi Afrizal (2012), adalah bahasa untuk menspesifikasi, memvisualisasi, membangun dan mendokumentasikan artifacts (bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak, artifact tersebut dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya.Selain itu UML adalah bahasa pemodelan yang menggunakan konsep orientasi object. UML dibuat oleh Grady Booch, James Rumbaugh, dan Ivar Jacobson di bawah bendera Rational Software Corp. UML menyediakan notasi-notasi yang membantu memodelkan sistem dari berbagai perspektif. UML tidak hanya digunakan dalam pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan.

2.5.1 Diagrams (diagram)

Suatu sistem yang kompleks harus dapat dipandang dari sudut yang berbeda-beda sehingga didapat pemahaman secara menyeluruh. Untuk keperluan tersebut UML menyediakan 9 jenis diagrams yang dapat dikelompokkan berdasarkan sifatnya; Statis atau Dinamis. Kesembilan diagram ini tidak mutlak digunakan atau dibuat sesuai kebutuhan. Pada pemodelan UML dimungkinkan menggunakan diagram lain sejauh diagram tersebut dapat membantu pemahaman mendalam tentang suatu sistem atau perangkat lunak. Berikut adalah diagram tersebut :

a. *Class Diagrams* (Statis)

Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, dan relasi-relasi.

1. *Class* Diagram menggambarkan interaksi antar kelas dalam sistem tersebut.
2. Pembuatan *Class* sama dengan pembuatan Objek-objek

b. *Activity Diagrams* (Dinamis)

Diagram ini adalah tipe khusus dari diagram state yang memperlihatkan aliran dari suatu aktifitas ke aktifitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi dalam suatu sistem dan memberi tekanan pada aliran kendali antar objek.

1. Memberikan gambaran ilustrasi alur dari setiap fungsi yang ada dalam sistem
2. Aliran dimulai dari suatu titik hingga ke titik akhir yang disepakati.

c. *Use-Case Diagrams* (Statis)

Dagram ini memperlihatkan himpunan *Use-Case* dan *actor-actor* (jenis khusus dari kelas). Diagram ini penting untuk mengorganisasi dan memodelkan perilaku dari suatu sistem yang dibutuhkan serta diharapkan pengguna.

1. *Use-Case* diagram menggambarkan hubungan *use-case* dengan *actor*.
2. *Use-case* merepresentasikan fungsi, kebutuhan dari perpektif *user*

3. *Actor* adalah orang atau sistem yang menerima atau memberikan informasi dari sistem ini

d. *Sequence Diagrams* (Dinamis)

Diagram *sequence* (urutan) adalah diagram interaksi yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu.

1. *Sequence* Diagram mengambarkan alur kerja dari fungsi-fungsi dalam sistem dengan *use-case* dimana didalamnya terdapat *actor*
2. *Actor* adalah orang atau sistem yang menerima atau memberikan informasi dari sistem ini.
3. Diagram ini sangat memperhatikan waktu/ terurut berdasarkan kejadian (*sequence*)

e. *Deployment Diagrams* (Statis)

Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*run-time*), memuat simpul-simpul atau node beserta komponen-komponen yang ada didalamnya. *Deployment Diagrams* berhubungan erat dengan diagram komponen dimana *Deployment Diagrams* memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing*).

1. Diagram *Deployment* menggambarkan bentuk *layout* secara fisik bentuk jaringan dan posisi komponen-komponen dari sistem
2. Pendekatan yang digunakan dalah pendekatan terhadap hasil implementasi/ program

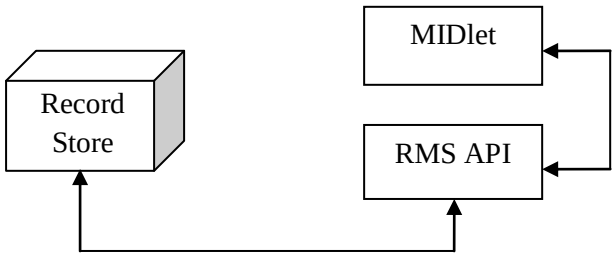
2.5.2 Tujuan UML

Tujuan UML diantaranya yaitu :

1. Memberikan model yang siap pakai, bahasa pemodelan visual yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah dan dimengerti secara umum.
2. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemrograman dan proses rekayasa.
3. Menyatukan praktek-praktek terbaik yang terdapat dalam pemodelan.

2.6 RMS (*Record Managemen System*)

Menurut Shalahuddin dan Rosa (2010) RMS (*Record Management System)* adalah kumpulan record , dan record di simpan sabagai array dari byte dalam sebuah record store. RMS (*Record Management System)* memiliki orientasi record basis data yang sederhana sehingga MIDlet dapat menyimpani informasi dan mengaksesnya. MIDlet yang berbeda dapat mengakses RMS (*Record Management System)* yang sama. Keterkaitan MIDlet, RMS (*Record Management System)* dan record store dapat dilihat pada gamabar di bawah ini.



Gambar 2.2 Keterkaitan MIDlet dan Record Store
Sumber : M.Shalahudin dan Rosa A.S (2010:122)

Mekanisme penyimpanan data pada MIDP telah tersedia secara persisten (tetap) dalam memori *device* yang bersangkutan. Data tersebut berupa kumpulan-kumpulan *record*. *Record* yang telah ditempatkan di dalam ruang penyimpanan dapat diambil kembali untuk kemudian digunakan sesuai kebutuhan. Ruang penyimpanan yang tetap (*persisten storage*) bersifat *non-volatile* (data tidak akan hilang walaupun program direstart atau *device* dimatikan) untuk menyimpan data dalam sebuah aplikasi.

Pada konsepnya, *record* yang disimpan dalam *record store* diakses berdasarkan recordId yang berupa *integer*. RecordId ini biasanya digunakan untuk mengakses *record* seperti fungsi indeks pada pengaksesan *array*. Pada saat data disimpan di dalam sebuah *record store*, data tersebut berada dalam bentuk sekumpulan larik dari tipe *byte (array byte).* Namun sebaliknya dalam pemrograman java, di sisi aplikasi, data harus selalu dalam bentuk objek. Sehingga untuk melakukan penyimpanan data dari aplikasi ke dalam *record store*, terlebih dulu dilakukan serialisasi dari bentuk objek ke bentuk *array byte*. Adapun untuk mengembalikan data dari *record store* ke aplikasi, juga perlu dilakukan deserialisasi dari bentuk *array byte* ke dalam bentuk objek. Dalam Record store MIDlet RMS API pemrograman MIDP, tidak didukung adanya fasilitas untuk serialisasi/deserialisasi untuk objek Java yang ada di dalam J2SE (*Java 2 Platform Standart Edition)*. Untuk itu dilakukan proses tersebut secara manual melalui kode-kode program yang ditulis, melalui kelas-kelas yang tersimpan dalam paket java.io.

Beberapa hal untuk melakukan proses serialisasi/ deserialisasi data di dalam pemrograman MIDP antara lain: *Pertama*, Kelas InputStream dan OutputStream merupakan kelas abstrak yang merepresentasikan stream data yang didefinisikan untuk method-method dasar guna keperluan baca/ tulis dari/ ke stream. *Kedua*, Kelas ByteArrayStream dan ByteArrayOutputStream merupakan subkelas dari InputStream dan OutputStream untuk melakukan baca/ tulis dari/ ke *byte array*. *Ketiga*, Kelas DataInputStream dan DataOutputStream untuk mengaitkan objek InputStream dan OutputStream yang ada. Kelas –kelas inii menyediakan method-method untuk proses baca/ tulis terhadap variable-variabel dengan tipe primitif maupun objek-objek *String* dari/ ke *byte* array ke dalam *stream* data.