BAB II

LANDASAN TEORI

2.1. Pelacakan Kiriman

http://kamusbahasaindonesia.org//pelacakan/22 mei 2012

Dalam kamus bahasa Indonesia pelacakan merupakan proses atau cara melacak atau mencari. Pelacakan juga dapat di definisikan sebagai teknik untuk mencari sesuatu. Didalam pencarian ada dua kemungkinan hasil yang didapat yaitu menemukan dan tidak ditemukan. Istilah kiriman yaitu sesuatu yang dikirimkan. Maka Pelacakan kiriman dapat didefinisikan sebagai proses pencarian sesuatu yang dikirim. Keberhasilan dan kualitas pencarian diukur dari empat cara, antara lain :

1. Kelengkapan

   Apakah algoritma pencarian menjamin untuk mendapatkan sebuah penyelesaian jika ada penyelesaian

2. Optimal

   Apakah algoritma pencarian akan mendapatkan penyelesaian optimal missal, penyelesaian dengan biaya lintasan minimum.

3. Kekomplekan waktu

   Berapa lama waktu yang digunakan untuk menyelesaikan permasalahan

4. Kekomplekan Ruang

   Berapa banyak memori yang dibutuhkan untuk melakukan pencarian.

2.2. Layanan Pos *Express*

Layanan Pos *Express* merupakan salah satu produk layanan PT. POS INDONESIA yang lebih *spesifik* melayani pengiriman kiriman pada kota-kota tertentu. Kiriman yang dikirim berupa barang dan surat/dokumen.

Adapun jenis layanan Pos *Express* diantaranya :

1. *Sameday Service*

   Jenis layanan Pos *Express* dimana kiriman tiba ditempat tujuan dihari yang sama ketika kiriman dikirim.

2. *Nextday Service*

   Jenis layanan Pos *Express* berupa kiriman hari ini, tiba ditujuan keesok hari nya.

2.2.1. End to End Service

End to End Service merupakan mata rantai nilai (*value chain*) proses operasi dari sipengirim/sipembayar sampai dengan sipenerima, baik penerima kiriman maupun penerima pembayaran. Adapun proses layanan operasi yaitu sebagai berikut :

1. Input merupakan pengeposan kiriman oleh si pengirim atau pembayaran diloket.
2. Proses merupakan pelaksanaan pemrosesan administrasi penerimaan kiriman diloket, sortir, pengantungan/tutupan, dan penyusunan kiriman pos (*grouping*), ataupun proses *transmitting* data ke unit kerja lain baik *internal* maupun *ekternal*.

3. Output merupakan pengiriman dengan menyerahkan kiriman pos kepada pihak pengangkut ataupun diterimanya data oleh unit lain.

2.2.2. Outgoing Management

Outgoing management merupakan suatu *method* pengaturan dan pengendalian *input*, proses dan *output* agar sesuai dengan standar-standar pengolahan *indoor process* serta dapat memenuhi parameter *indoor process* yaitu kapasitas, waktu dan sasaran (akurasi). Adapun tahapan proses utama operasi *outgoing management* yakni :

1. Penerimaan dari Pengirim

   Pada proses penerimaan dari pengirim hal terpenting yang harus dilakukan adalah menetapkan jam buka atau tutup loket sebagai media informasi kepada para pelanggan atau masyarakat pengguna layanan jasa pos.

2. Pemrosesan

   Pelaksanaan pemrosesan kiriman akan terkait dengan skala prioritas pengolahan sesuai dengan jenis kiriman yang bertalian (standar/prioritas). Pemrosesan terdiri dari aktifitas sortir dan pengantungan atau tutupan. Ketentuan yang harus dilaksanakan pada sortir dan pengantungan adalah

   a. Sortir

      Kiriman Standar yaitu dimasukan dalam lubang sortir kantor SPP/KSD, khusus untuk kiriman yang ditujukan untuk wilayah Jawa dan Sumatera sortiran dilakukan mengacu pada kelompok Kp *inbound* berdasarkan kantor SPP/KSD tujuan sesuai dengan

penetapan jaringan *Next Day Delivery* (NDD) di Jawa serta pada jaringan *Time Certain Delivery* (TCD) di Sumatera. Kiriman Prioritas yaitu dimasukan dalam lubang sortir Kantor Tukar dalam negeri.

b. Pengantungan/Tutupan

Kiriman tandar (dn/n) dilakukan secara curah yang dibendel dengan carik ikatan N14a(dn) dan ditutup dalam kantung dengan tujuan kantor SPP/KSD atau Kantor Tukar dalam negeri yang bertalian. Kiriman Prioritas (dn) ditutup langsung ke kanor tujuan yang bertalian, dengan memperhatikan kantor tujuan telah diikutkan dalam jaringan pelayanan prioritas. Kiriman prioritas (In) dilakukan secara curah yang dibendel dengan cariki ikatab dan ditutup dalam kantung yang ditujukan kepada Kantor Tukar dan yang bertalian.

3. Penyusunan Kiriman Pos (grouping)

Grouping kiriman pos didalam alat angkut dilakukan berdasarkan prioritas pelaksanaan bongkar atau muat kiriman pos sesuai dengan jalur distribusi kiriman pos.

4. Pengiriman

Proses akhir dari *outgoing managenent* adalah pengiriman yaitu proses penyerahan kiriman pos kepada pengangkutan untuk dikirimkan ke kantor tujuan. Pada proses pengiriman hal terpenting yang harus diketahui adalah jam keberangkatan dari alat angkutan (darat, laut dan udara) baik melalui

jalur primer maupun jalur sekunder. Pengiriman kiriman pos melalui alat angkutan primer harus dipastikan sesuai dengan jadwal keberangkatan yang telah ditetapkan, serta kecepatan pola pengiriman sesuai dengan jalur distribusi kirim pos.

2.3. Aplikasi *Mobile*

http://agusbarupunyablog.blogspot.com/pengertian-aplikasi-mobile/22Maret 2012

Aplikasi *mobile* merupakan aplikasi yang dapat digunakan walaupun pengguna berpindah dengan mudah dari satu tempat ketempat lain tanpa terjadi pemutusan atau terputusnya komunikasi. Aplikasi *mobile* hanya dapat diakses oleh perangkat *mobile* seperti, *smartphone* dan *tablet PC*. Karakteristik perangkat *mobile*, sebagai berikut:

1. Ukuran yang Kecil

   Perangkat *mobile* memiliki ukuran yang kecil, konsumen menginginkan perangkat yang terkecil untuk kenyamanan dan mobilitas mereka.

2. *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.

3. Daya proses yang terbatas

   Sistem *mobile* tidaklah setangguh desktop. Ukuran teknologi dan biaya adalah beberapa faktor yang mempengaruhi status dari sumber daya ini.

4. Mengonsumsi daya yang rendah

   Perangkat *mobile* menghabiskan sedikit daya dibandingkan dengan mesin desktop. Daya yang disedikan dibatasi oleh baterai-baterai.

5. 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.

6. Konektivitas yang terbatas

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

7. Masa hidup yang pendek

   Perangkat-perangkat konsumen ini menyala dalam hitungan detik kebanyakan dari mereka selalu menyala.

2.4. Android

Android adalah sistem operasi yang berbasis *linux* untuk telepon *seluler* seperti telepon pintar dan komputer *tablet.* Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak.

Menurut **Jeff Lessard** dan **Gary C.Kessler** dalam jurnalnya berjudul ***Android Forensic : Simplifying Cell Phone Examinations***,

"Android is an operating system (OS) developed by theOpen Handset Alliance (OHA). The Alliance is a coalition of more than 50 mobile technology companies ranging from handset manufactures and service providers to semiconductor manufacturers and software developers, including Acer, ARM, Google, eBay, HTC, Intel, LG Electronics, Qualcomm, Sprint, and T-Mobile. The stated goal of the OHA is to "accelerate innovation in mobile and offer consumers a richer, less expensive, and better mobile experience" (OHA, 2009, n.p.)."

Android merupakan sistem operasi *open platform* yang dikembangkan oleh *Open Handset Alliance* (OHA). Dengan tujuan membuat sebuah standar terbuka untuk perangkat bergerak *mobile device.*

2.4.1. Sejarah Android

Android Inc adalah sebuah perusahaan *software* kecil yang didirikan pada bulan Oktober 2003 di Palo Alto, California, USA. Didirikan oleh beberapa senior di beberapa perusahaan yang berbasis IT & *Communication* Andy Rubin, Rich Miner, Nick Sears dan Chris White. Menurut Rubin, Android Inc didirikan untuk mewujudkan *mobile device* yang lebih peka terhadap lokasi dan *preferensi* pemilik. Dengan kata lain, Android Inc ingin mewujudkan *mobile device* yang lebih mengerti pemiliknya. Pada bulan nopember 2007, terbentuklan *Open Handset Alliance* yang merupakan *konsorsium* dari beberapa perusahaan, *Broadcom Corporation, Google, HTC, Intel, LG, Marvell Technology Group, Motorola, Nvidia, Qualcomm, Samsung Electronics, Sprint Nextel, T-Mobile dan Texas Instruments.* Mereka sepakat untuk membuat *open standart* bagi *mobile phone.* Pada hari yang sama, mereka mengumumkan produk pertama mereka yaitu *Android* yang berbasis *Linux kernel versi 2.6.* Bulan Desember 2008,

bergabunglah 14 perusahaan lainnya yaitu ARM *Holdings*, Atheros *Communications*, Asustek *Computer* Inc, Garmin Ltd, PacketVideo, Softbank, Sony Ericsson, Toshiba Corp dan Vodafone Group Plc. Seiring dengan perkembangan android, Android mempunyai tujuh *versi* sistem operasi diantaranya versi 1.1, versi 1.5 cupcake, versi 1.6 donut, versi 2.0/2.1 Éclair, versi 2.2 Froyo (*Frozen Yoghurt*), versi 2.3 Gingerbread, versi 3.0 Honeycomb sebagai berikut :

1. Android versi 1.1

   Pada 9 Maret 2009, Google merilis Android versi 1.1. Android *versi* ini dilengkapi dengan pembaruan estetis pada aplikasi, jam *alarm*, *voice search*, pengiriman pesan dengan *Gmail*, dan pemberitahuan *email*.

2. Android versi 1.5 Cupcake

   Pada pertengahan Mei 2009, *Google* kembali merilis telepon *seluler* dengan menggunakan Android dan SDK (*Software Development Kit*) dengan versi 1.5 Cupcake. Terdapat beberapa pembaruan termasuk juga penambahan beberapa *fitur* dalam *seluler versi* ini yakni kemampuan merekam dan menonton *video* dengan modus kamera, mengunggah *video* ke*Youtube* dan gambar ke picasa langsung dari telepon, dukungan *Bluetooth* A2DP, Kemampuan terhubung secara otomatis ke *headset Bluetooth*, animasi layar, dan keyboard pada layar yang dapat disesuaikan dengan sistem.

3. Android versi 1.6 Donut

   Android versi 1.6 donut dirilis pada september dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai *indikator* dan kontrol *applet* VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus kamera, camcorder dan galeri yang dintegrasikan CDMA / EVDO, 802.1x, VPN, Gestures, dan *Text-to-speech engine* kemampuan dial kontak, teknologi *text to change speech* dan tidak tersedia pada semua ponsel pengadaan resolusi VWGA.

4. Android versi 2.0 / 2.1 Éclair

   Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 Eclair, perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan Google *Maps* 3.1.2, perubahan UI dengan *browser* baru dan dukungan HTML5, daftar kontak yang baru, dukungan *flash* untuk kamera 3,2 MP. Dengan semakin berkembangnya dan semakin bertambahnya jumlah *handset* Android, semakin banyak pihak ketiga yang berminat untuk menyalurkan aplikasi mereka kepada sistem operasi Android. Aplikasi terkenal yang diubah ke dalam sistem operasi Android adalah *Shazam, Backgrounds,* dan *WeatherBug.* Sistem operasi Android dalam situs *internet* juga dianggap penting untuk menciptakan aplikasi Android asli, contohnya oleh *MySpace* dan *Facebook.*

5. Android versi 2.2 Froyo (Frozen Yoghurt)

   Pada 20 Mei 2010, Android versi 2.2 Froyo diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan *Adobe Flash 10.1*, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, *intergrasi* V8 *JavaScript engine* yang dipakai Google Chrome yang mempercepat kemampuan *rendering* pada *browser*, pemasangan aplikasi dalam SD *Card*, kemampuan WiFi Hotspot *portabel*, dan kemampuan *auto update* dalam aplikasi Android *market.*

6. Android versi 2.3 Gingerbread

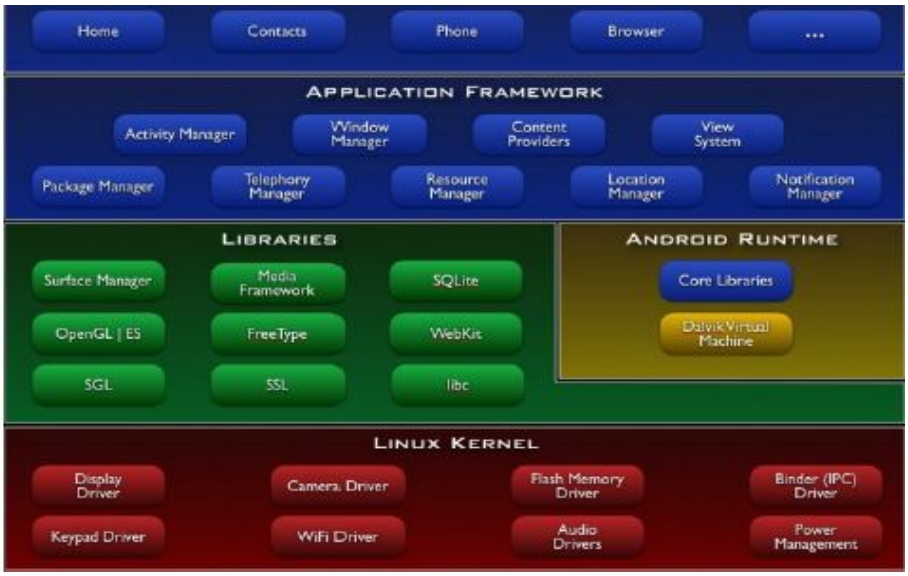
   Pada 6 Desember 2010, Android versi 2.3 Gingerbread diluncurkan. Perubahan-perubahan umum yang didapat dari Android versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antar muka (*User Interface*) didesain ulang, dukungan format *video* VP8 dan WebM, efek audio baru (*reverb, equalization, headphone virtualization,* dan *bass boost*) dukungan kemampuan *Near Field Communication* (NFC), dan dukungan jumlah kamera yang lebih dari satu.

7. Android versi 3.0 Honeycomb

   Android Honeycomb dirancang khusus untuk *tablet*. Android versi ini mendukung ukuran layar yang lebih besar. *User Interface* pada Honeycomb juga berbeda karena sudah didesain untuk *tablet.* Honeycomb juga mendukung *multi prosesor* dan juga akselerasi perangkat keras (*hardware*) untuk grafis.

2.4.2. Arsitektur Android

Arsitektur Android menunjukkan komponen-komponen utama yang terdapat pada sistem operasi Android. Berikut ini arsitektur android yang digambarkan pada **Gambar 2.1**



Gambar 2.1 Arsitektur Android
Sumber : **Small Scale Digital Device Forensics Journal Vol. 4, No.1, September 2010, Issn# 1941-6164**

Menurut Jeff Lessard dan **Gary C.Kessler** dalam jurnalnya berjudul ***Android Forensic:Simplifying Cell Phone Examinations,*** menyatakan :

"*Android OS builds are based on the Linux 2.6 kernel. When running on a hard drive, the Linux system device defaults to the first physical hard drive, or /dev/hd0. In addition, Linux only understands character and block devices,such as keyboards and disk drives, respectively. With Linux on flash, however, a Flash Transition layer provides the system device functionality. A Memory Technology Device (MTD) is needed to provide an interface between the Linux OS and the physical flash device because flash memory devices are not seen as character or block devices* (Dedekind, 2009)**"**.

1. Linux Kernel

   Android dibangun di atas kernel Linux 2.6. Namun secara keseluruhan android bukanlah linux, karena dalam android tidak terdapat paket standar yang dimiliki oleh linux lainnya. Linux merupakan sistem operasi

terbuka yang handal dalam manajemen memori dan proses. Oleh karenanya pada android hanya terdapat beberapa servis yang diperlukan seperti keamanan, manajemen memori, manajemen proses, jaringan dan *driver*. Kernel linux menyediakan *driver* layar, kamera, keypad, WiFi, Flash Memory, audio, dan IPC (*Interprocess Communication*) untuk mengatur aplikasi dan lubang keamanan.

2. Libraries

   Android menggunakan beberapa paket pustaka yang terdapat pada C/C++ dengan standar Berkeley Software Distribution (BSD) hanya setengah dari yang aslinya untuk tertanam pada kernel Linux. Beberapa pustaka diantaranya:

   a. Media Library untuk memutar dan merekam berbagai macam format audio dan video.
   b. Surface Manager untuk mengatur hak akses layer dari berbagai aplikasi.
   c. Graphic Library termasuk didalamnya SGL dan OpenGL, untuk tampilan 2D dan 3D.
   d. SQLite untuk mengatur relasi database yang digunakan pada aplikasi.
   e. SSl dan WebKit untuk browser dan keamanan internet.

   Pustaka-pustaka tersebut bukanlah aplikasi yang berjalan sendiri, namun hanya dapat digunakan oleh program yang berada di level atasnya. Sejak versi Android 1.5, pengembang dapat membuat dan

menggunakan pustaka sendiri menggunakan Native Development Toolkit (NDK).

3. Android Runtime

Pada android tertanam paket pustaka inti yang menyediakan sebagian besar fungsi android. Inilah yang membedakan Android dibandingkan dengan sistem operasi lain yang juga mengimplementasikan Linux. Android Runtime merupakan mesin virtual yang membuat aplikasi android menjadi lebih tangguh dengan paket pustaka yang telah ada.

a. Pustaka Inti, android dikembangkan melalui bahasa pemrograman Java, tapi Android Runtime bukanlah mesin virtual Java. Pustaka inti android menyediakan hampir semua fungsi yang terdapat pada pustaka Java serta beberapa pustaka khusus android.
b. Mesin Virtual Dalvik, Dalvik merupakan sebuah mesin virtual yang dikembangkan oleh Dan Bornstein yang terinspirasi dari nama sebuah perkampungan yang berada di Iceland. Dalvik hanyalah interpreter mesin virtual yang mengeksekusi file dalam format Dalvik Executable (*.dex). Dengan format ini Dalvik akan mengoptimalkan efisiensi penyimpanan dan pengalamatan memori pada file yang dieksekusi. Dalvik berjalan di atas kernel Linux 2.6, dengan fungsi dasar seperti threading dan manajemen memori yang terbatas. [Nicolas Gramlich, Andbook, anddev.org].

4. Aplikasi Framework

Kerangka aplikasi menyediakan kelas-kelas yang dapat digunakan untuk mengembangkan aplikasi android. Selain itu juga menyediakan abstraksi generik untuk mengakses perangkat, serta mengatur tampilan user interface dan sumber daya aplikasi. Bagian terpenting dalam kerangka aplikasi android adalah sebagai berikut :

a. *Activity Manager*, berfungsi untuk mengontrol siklus hidup aplikasi dan menjaga keadaan ”Backstack“ untuk navigasi penggunaan.
b. *Content Providers*, berfungsi untuk merangkum data yang memungkinkan digunakan oleh aplikasi lainnya, seperti daftar nama.
c. *Resuource Manager*, untuk mengatur sumber daya yang ada dalam program. Serta menyediakan akses sumber daya diluar kode program, seperti karakter, grafik, dan file layout.
d. *Location Manager*, berfungsi untuk memberikan informasi detail mengenai lokasi perangkat android berada.
e. *Notification Manager*, mencakup berbagai macam peringatan seperti, pesan masuk, janji, dan lain sebagainya yang akan ditampilkan pada status bar.

5. Application Layer

Puncak dari diagram arsitektur android adalah lapisan aplikasi dan widget. Lapisan aplikasi merupakan lapisan yang paling tampak pada pengguna ketika menjalankan program. Pengguna hanya akan melihat

program ketika digunakan tanpa mengetahui proses yang terjadi dibalik lapisan aplikasi. Lapisan ini berjalan dalam Android runtime dengan menggunakan kelas dan service yang tersedia pada framework aplikasi.Lapisan aplikasi android sangat berbeda dibandingkan dengan sistem operasi lainnya. Pada android semua aplikasi, baik aplikasi inti (native) maupun aplikasi pihak ketiga berjalan diatas lapisan aplikasi dengan menggunakan pustaka API (*Application Programming Interface*) yang sama.

2.4.3. Komponen Android

1. Service

   Service diproses tidak terlihat, memperbarui sumber data dan menampilkan notifikasi. *Service* digunakan untuk melakukan pengolahan data yang perlu terus diproses, bahkan ketika *Activity* tidak aktif atau tidak tampak.

2. Intents

   Intens merupakan sebuah mekanisme untuk menggambarkan tindakan tertentu, seperti memilih foto, menampilkan halaman web, dan lain sebagainya. Intents tidak selalu dimulai dengan menjalankan aplikasi, namun juga digunakan oleh sistem untuk memberitahukan ke aplikasi bila terjadi suatu hal, misal pesan masuk. Intents dapat eksplisit atau implisit, contohnya jika suatu aplikasi ingin menampilkan URL, sistem akan menentukan komponen apa yang dibutuhkan oleh Intents tersebut.

3. Broadcast Receivers

Broadcast Receivers merupakan komponen yang sebenarnya tidak melakukan apa-apa kecuali menerima dan bereaksi menyampaikan pemberitahuan. Sebagian besar *Broadcast* berasal dari sistem misalnya, Batre sudah hampir habis, informasi zona waktu telah berubah, atau pengguna telah merubah bahasa *default* pada perangkat. *Broadcast Receivers* dapat menggunakan *Notification Manager* untuk memberitahukan sesuatu kepada pengguna.

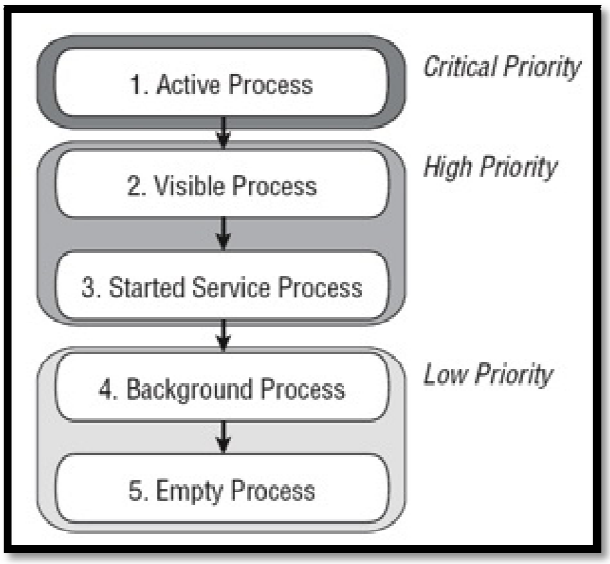
4. Content Providers

Content Providers digunakan untuk mengelola dan berbagi database. Data dapat disimpan dalam *file* sistem, dalam database SQLite, atau dengan cara lain yang pada prinsipnya sama. Dengan adanya *Content Provider* memungkinkan antar aplikasi untuk saling berbagi data. Komponen ini sangat berguna ketika sebuah aplikasi membutuhkan data dari aplikasi lain, sehingga mudah dalam penerapannya.

2.4.4. Sirklus Hidup Android

Dikelola oleh sistem, berdasarkan kebutuhan pengguna, sumberdaya yang tersedia, dan sebagainya. Android menjalankan setiap aplikasi dalam proses secara terpisah, yang masing-masing memliki mesin *virtual* pengolah sendiri, dengan ini melindungi penggunaan memori pada aplikasi. Selain itu juga android dapat mengontrol aplikasi mana yang layak menjadi prioritas utama. Karenanya android sangat *sensitive* dengan siklus hidup aplikasi dan komponen-

komponennya. Perlu adanya penanganan terhadap setiap kondisi agar aplikasi menjadi stabil.



Gambar 2.2 Prioritas Proses
Sumber : http://riyan214.wordpress.com

1. *Active Processes*

Aplikasi hosting dengan komponen saat berinteraksi dengan user. Ini adalah proses Android yang sedang mencoba untuk tetap *responsif* dengan *reclaiming* sumber. Umumnya proses tersebut sangat sedikit, dan mereka akan dieksekusi hanya sebagai pilihan terakhir. Proses *Aktive* meliputi:

a. *Active*

Setiap *activity* yang berada di awal, maka dia akan terlihat dan menerima input dari pengguna. Android akan berusaha membuat *activity* aplikasi untuk tetap hidup dan akan menghentikan *activity* yang berada di akhir jika diperlukan. Ketika *activity* sedang aktif, maka yang lainnya akan *waiting*.

b. *Pause*

Dalam beberapa *activity* akan terlihat tapi tidak pada kondisinya. Ini terjadi jika *activity* tidak *fullscreen* dan transparan pada layar. Ketika *activity* dalam keadaan *paused*, akan terlihat *active* namun tidak dapat menerima input dari pengguna. Dalam kasus ekstrim, android akan menghentikan *activity* dalam keadaan *paused.*

c. *Stopped*

Activity yang tidak terlihat. *Activity* tetap berada dalam memori dengan informasi yang ada. Tetapi akan tetap dieksekusi awal oleh sistem ketika sistem membutuhkan sumber yang lebih. Saat *activity* dalam kondisi *stopped* maka data disimpan. Ketika *activity* ditutup, maka akan menjadi *aktive* didalam sistem.

d. *Inactive*

Activity telah dihentikan dan sebelum dieksekusi. *Activity* telah ditutup dari sistem sehingga perlu di *restart* ulang agar dapat digunakan kembali. Proses ini dijalankan dalam manajer memori android. Android akan menutup aplikasi jika *activity inactive*, kemudian *stopped activity.*

2. *Visible Processes*

Visible tidak aktif tetapi aktivitas hosting tetap terlihat. Seperti namanya, Proses terlihat , tetapi tidak di latar depan atau menanggapi

perintah user. Hal ini terjadi ketika sebuah Aktivitas hanya sebagian yang tertutup.

3. *Started Service Processes*

   Proses layanan-layanan hosting yang telah dimulai. Layanan mendukung proses berkelanjutan yang harus melanjutkan tanpa antarmuka. Karena Jasa tidak berinteraksi langsung dengan *user*, langsung menerima prioritas yang sedikit lebih rendah daripada aktivitas yang terlihat.

4. *Background Processes*

   Proses masih dianggap sebagai proses latar depan dan tidak akan dianggap proses background meskipun sudah mulai dieksekusi. Umumnya ada sejumlah besar proses *background* Android yang akan mengeksekusi akhir setelah yang awal dieksekusi untuk mendapatkan sumber proses latar depan.

5. *Empty Processes*

   Untuk meningkatkan kinerja sistem secara keseluruhan, Android sering mempertahankan aplikasi dalam memori, setelah mencapai akhir eksekusi proses. Android mempertahankan cache untuk meningkatkan waktu *start-up* aplikasi ketika kembali diluncurkan. Proses *Rou-tinely* ini dieksekusi sesuai kebutuhan.

2.4.5. Tipe Aplikasi Android

Terdapat tiga kategori aplikasi pada android :

1. *Foreground Activity*

   Aplikasi yang hanya dapat dijalankan jika tampil pada layar dan tetap *efektif* walaupun tidak terlihat. Aplikasi dengan tipe ini pasti mempertimbangkan siklus hidup *activity*, sehingga perpindahan antar *activity* dapat berlangsung dengan lancar.

2. *Background Service*

   Aplikasi yang memiliki interaksi terbatas dengan *user*, selain dari pengaturan konfigurasi, semua dari prosesnya tidak tidak tampak pada layar. Contohnya aplikasi penyaringan panggilan atau sms auto respon.

3. *Intermittent Activity*

   Aplikasi yang masih membutuhkan beberapa masukkan dari pengguna, namun sebagian sangat *efektif* jika dijalankan di *background* dan jika diperlukan akan memberi tahu pengguna tentang kondisi tertentu. Contohnya pemutar musik. Untuk aplikasi yang kompleks akan sulit untuk menentukan kategori aplikasi tersebut apalagi aplikasi memiliki ciri-ciri dari semua kategori. Oleh karenanya perlu pertimbangan bagaimana aplikasi tersebut digunakan dan menentukan kategori aplikasi yang sesuai.

2.4.6. Keunggulan Android

1. Biaya yang rendah untuk memulai mengembangkan Aplikasi, SDK *Tools* bebas untuk di *download* dan multi *platform*. Biaya yang terjangkau untuk mendistribusikan aplikasi di Android Market.

2. Model Distribusi yang terbuka. Pengembang bebas mendistribusikan aplikasi bisa melalui Google Android *Market* atau melalui saluran distribusi yang lain seperti Amazon App Store, dan lain-lain.
3. *Multi-platform*. Ada berbagai macam perangkat keras didukung oleh OS Android, termasuk telepon yang berbeda dan komputer tablet. Pengembangan *platform* dapat dijalankan pada Windows, Mac OS atau Linux.
4. Dukungan *multi-carrier*. Sejumlah besar operator telekomunikasi saat ini menawarkan dukungan terhadap ponsel Android.

2.5. Metode Pengembangan Sistem

Dalam pembangunan apapun khususnya pembangunan sistem dalam pelaksanaannya pada awalnya perlu mengidentifikasi masalah yang ada. Selanjutnya ditentukan metode yang sesuai untuk menyelesaikan masalah yang timbul. Metode yang dipakai untuk penyelesaian masalah disebut metode pengembangan sistem. Metode pengembangan sistem terdiri dari beberapa metode diantaranya RUP (*Rational Unified Process)*.

2.5.1. RUP (*Rational Unified Process)*

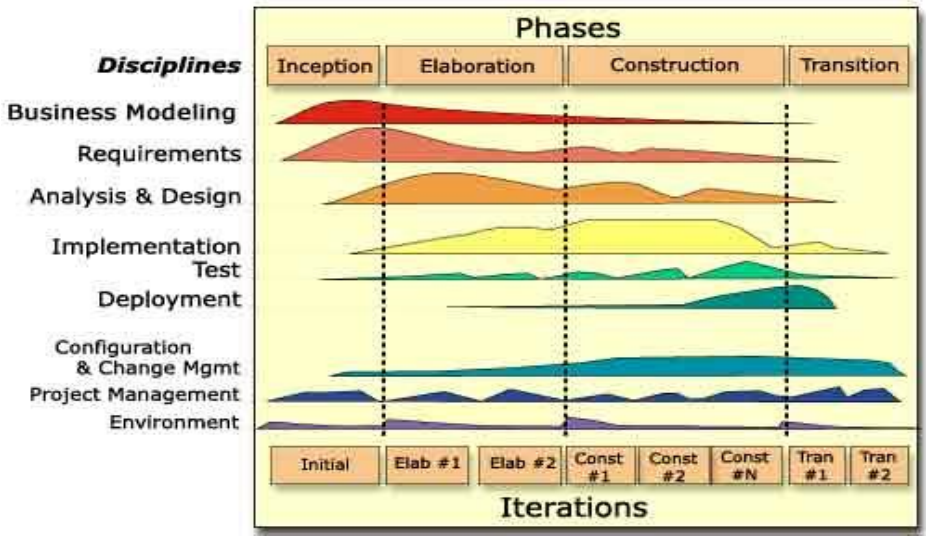
http://id.wikipedia.org/wiki/*RUP*/18 maret 2012

RUP singkatan dari *Rational Unified Process,* adalah suatu kerangka kerja proses pengembangan perangkat lunak *iteratif* yang dibuat oleh *Rational Software*, suatu *divisi* dari IBM sejak 2003. RUP bukanlah suatu proses tunggal dengan aturan yang konkrit, melainkan suatu kerangka proses yang dapat diadaptasi dan dimaksudkan untuk disesuaikan oleh organisasi pengembang dan

tim proyek perangkat lunak yang akan memilih elemen proses sesuai dengan kebutuhan mereka.

RUP menggunakan konsep *object oriented*, dengan aktifitas yang berfokus pada pengembangan model dengan menggunakan *Unified Model Language* (UML). RUP memiliki dimensi pertama dan dimensi kedua sebagai berikut :

1. Dimensi pertama digambarkan secara *horizontal*. Dimensi ini mewakili aspek-aspek dinamis dari pengembangan perangkat lunak. Aspek ini dijabarkan dalam tahapan pengembangan atau *fase*. Setiap *fase* akan memiliki suatu *major milestone* yang menandakan akhir dari awal dari *fase* selanjutnya. Setiap *fase* dapat berdiri dari satu beberapa iterasi. Dimensi ini terdiri atas *Inception, Elaboration, Construction,* dan *Transition.*
2. Dimensi kedua digambarkan secara *vertikal.* Dimensi ini mewakili aspek-aspek statis dari proses pengembangan perangkat lunak yang dikelompokkan ke dalam beberapa disiplin. Proses pengembangan perangkat lunak yang dijelaskan kedalam beberapa disiplin terdiri dari empat elemen penting, yakni *who is doing*, *what*, *how* dan *when*. *Business Modeling, Requirement, Analysis and Design, Implementation, Test, Deployment, Configuration* dan *Change Manegement, Project Management, Environtment.*



Gambar 2.3 Grafik Dimensi Pertama
Sumber* : *Software Engineering Methodology Rational Unified Process

Berdasarkan pada **Gambar 2.3** diatas, menjelaskan mengenai *fase-fase* pada *Rational Unified Process* yang termasuk ke dalam dimensi pertama dari RUP sebagai berikut :

1. *Inception,* merupakan tahap untuk mengidentifikasi sistem yang akan dikembangkan. Aktivitas yang dilakukan pada tahap ini antara lain mencakup analisis sistem *eksisting,* perumusan sistem target, penentuan arsitektur global target, identifikasi kebutuhan,perumusan persyaratan (fungsional, performansi, keamanan, GUI, dll) perumusan kebutuhan pengujian (*level* unit, integrasi, sistem, performansi, fungsionalitas, keamanan, dll) pemodelan diagram UML (diagram *use case* dan *activity*), dan pembuatan dokumentasi
2. *Elaboration,* merupakan tahap untuk melakukan desain secara lengkap berdasarkan hasil analisis di tahap *inception*. *Aktivitas* yang dilakukan pada tahap ini antara lain mencakup pembuatan desain arsitektur subsistem (*architecture pattern*), desain komponen sistem, desain format

data (protokol komunikasi), desain database, desain antarmuka/tampilan, desain peta aliran tampilan, penentuan *design pattern* yang digunakan, pemodelan diagram UML (diagram *sequence*, *class*, *component*, *deployment*, dll.), dan pembuatan dokumentasi.

3. *Construction*, merupakan tahap untuk mengimplementasikan hasil desain dan melakukan pengujian hasil implementasi. Pada tahap awal *construction*, ada baiknya dilakukan pemeriksaan ulang hasil analisis dan desain, terutama desain pada domain perilaku (diagram *sequence*) dan domain struktural (diagram *class*, *component*, *deployment*). Apabila disain yang dibuat telah sesuai dengan analisis sistem, maka implementasi dengan bahasa pemrogramanan tertentu dapat dilakukan. Aktivitas yang dilakukan pada tahap ini antara lain mencakup pengujian hasil analisis dan disain (misal menggunakan *Class Responsibility Collaborator* untuk kasus pemrograman berorientasi obyek) pendataan kebutuhan implementasi lengkap (berpedoman pada identifikasi kebutuhan ditahap analisis), penentuan *coding pattern* yang digunakan, pembuatan program, pengujian, optimasi program, pendataan berbagai kemungkinan pengembangan atau perbaikan lebih lanjut, dan pembuatan dokumentasi.
4. *Transition*, merupakan tahap untuk menyerahkan sistem aplikasi ke konsumen (*roll-out*), yang umumnya mencakup pelaksanaan pelatihan kepada pengguna dan testing beta aplikasi terhadap ekspetasi pengguna.

Adapun kelebihan dari *Rational Unified Process* (RUP) diantaranya sebagai berikut :

1. Menyediakan akses yang mudah terhadap pengetahuan dasar bagi anggota tim.
2. Menyediakan petunjuk bagaimana menggunakan UML secara *efektif*.
3. Mendukung proses pengulangan dalam pengembangan *software.*
4. Memungkinkan adanya penambahan-penambahan pada proses.
5. Memungkinkan untuk secara sistematis mengontrol perubahan-perubahan yang terjadi pada *software* selama proses pengembangannya.
6. Memungkinkan untuk menjalankan *test case* dengan menggunakan *Rational Test Manager Tool.*

Sedang kelemahan *Rational Unified Process* (RUP) diantaranya yaitu metodologi ini hanya dapat digunakan pada penegembangan perangkat lunak yang berorientasi objek dengan berfokus pada UML (*Unified Modeling Language).*

2.6. Metode Pendekatan Sistem

Pada umumnya ada dua metode pendekatan sistem informasi yaitu pendekatan beriontasi objek dan pendekatan terstruktur. Untuk pendekatan objek, dalam melakukan pemecahan suatu masalah tidak dilihat bagaimana cara menyelesaikan suatu masalah tersebut tetapi objek-objek apa yang dapat melakukan pemecahan masalah tersebut. Pendekatan ini memungkinkan pembangunan *software* dengan cepat, sehingga dapat segera masuk ke pasaran dan *kompetitif*. Sistem yang dihasilkan sangat *fleksibel* dan mudah dalam

pemeliharaan. Sedangkan untuk pengembangan terstruktur, menggunakan prosedur atau tata cara yang teratur untuk mengoperasikan data struktur. Pendekatan ini, membuat berat pengguna pada dekomposisi fungsional.

2.6.1. UML (*Unified Modeling Language)*

Menurut Bambang Hariyanto. Ir.,MT. (2004:17), UML (*Unified Modeling Language)* adalah bahasa grafis untuk mendokumentasikan, menspeksifikasikan, dan membangun sistem perangkat lunak.UML berorientasi objek, menerapkan banyak *level* abstraksi, tidak bergantung proses pengembangan, tidak bergantung bahasa dan teknologi, pemanduan beberapa notasi di beragam metodologi, usaha bersama dari banyak pihak, didukung oleh kakas-kakas yang diintegrasikan lewat XML (XMI). Standar UML dikelola oleh OMG (Objek Management Group)

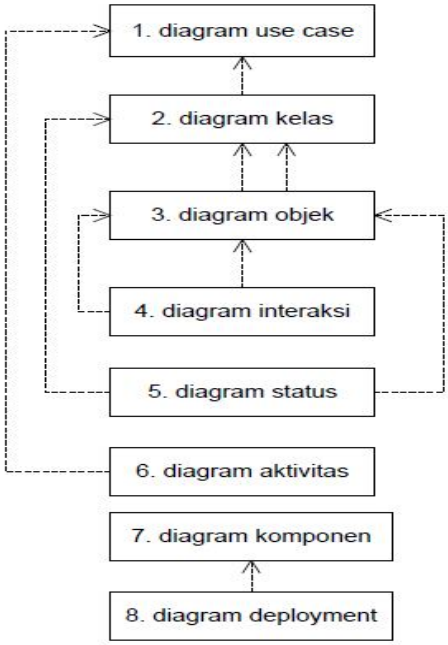
Adapun Tujuan UML (*Unified Modeling Language)* diantaranya :

1. Menyediakan bahasa pemograman *visual* yang *ekspresif* dan siap pakai untuk mengembangakan dan pertukaran model-model yang berarti.
2. Menyediakan mekanisame perluasan dan spesialisasi untuk memperluas konsep-konsep ini
3. Mendukung speksifikasi independen bahasa pemograman dan proses pengembangan tertentu
4. Menyediakan basis formal untuk pemahaman bahasa pemodelan
5. Mendorong pertumbuhan pasar kakas beriontasi objek
6. Mendukung konsep-konsep pengembangan level lebih tinggi seperti komponen, kolaborasi, framework, dan pattern.

Aturan-aturan pada UML bisa diklasifikasikan ke dalam:

1. *Syntactic* : rincian aspek dan kombinasi aturan;
2. *Semantic* : rincian arti dari simbol, secara individual dan dalam konteks
3. *Pragmatic* : panduan bagaimana menggunakan bahasanya (maksuk dari simbol).

Konsep yang paling penting dalam pemahaman UML adalah arsitektur UML, notasi (diagram), mekanisme ekstensi dan pembatas. Berikut adalah keterkaitan diagram-diagram pada UML beserta urutan pembuatannya.



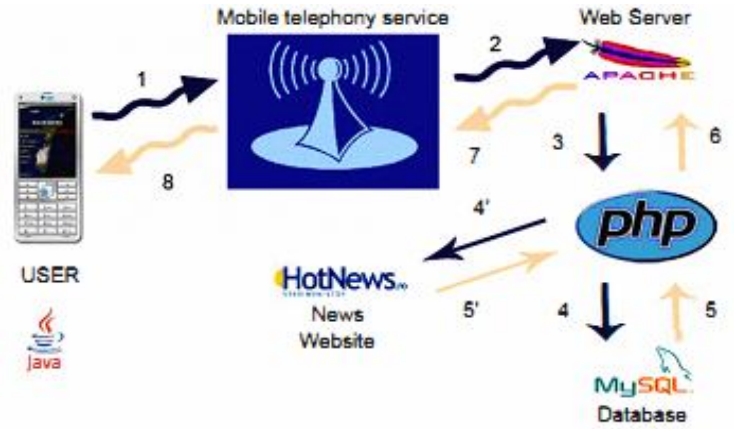
Gambar 2.4 Keterkaitan Diagram-Diagram UML
***Sumber* : Analisis dan Desain Sistem Informasi, Ardhian Agung Yulianto dkk**

2.7. Konsep *Client-Server*

Model *client-server* adalah sebuah model komputer yang bertindak sebagai aplikasi terdistribusi yang membagi beban tugas antara penyedia sumber daya

atau layanan yang disebut *server*, dan pemohon layanan yang disebut *client.* Seringkali *client* dan *server* melalui jaringan komputer pada *hardware* terpisah, tetapi keduanya berada di dalam sistem. Mesin *server* sebagai penyedia layanan (*host*) yang menjalankan satu atau lebih program *server* yang membagi sumber daya mereka dengan *client*. *Client* tidak membagi sumber daya yang dimilikinya, tetapi memohon *content* yang dimiliki *server* atau fungsi yang dimiliki *server.*

Karakteristik *client/server* menggambarkan hubungan program kerjasama di dalam sebuah aplikasi. Komponen *server* menyediakan sebuah fungsi atau layanan kepada satu atau beberapa *client.* Model *client-server* pada aplikasi *mobile* secara umum dapat dilihat pada **Gambar 2.6.**



Gambar 2.5 ***A simplified informational flow of the overall mobile application***
Sumber: Mobile Application for News and Interactive Services – L.Ashwin Kumar – Vol.2 No.1, January 2012 – ISSN 2225-7217

2.7.1. Karakteristik *Client-Server*

Adapun arakteristik umum *client* sebagai berikut :

1. Pasif
2. Menunggu *request*

3. Menerima *request*, memproses mereka dan mengirimkan balasan berupa *service.*

Sedangkan karakteristik *server* antara lain :

1. Aktif
2. Mengirim *request*
3. Mengirim *request* menunggu dan menerima balasan dari *server.*

2.7.2. Keuntungan dan Kerugian Arsitektur *Client-Server*

1. Semua data disimpan pada *server*, sehingga manajemen sekuritas penuh berada pada *server,* hal ini menjadi baik dibandingkan dengan manajemen sekuritas data yang tersebar, *server* menjamin siapa-siapa saja yang berhak mengakses data.
2. Oleh karena *storage* terpusat, proses *update* data menjadi mudah untuk dilakukan daripada melakukan *update* data yang tersebar. Pada kasus *peer to peer*, update data dilakukan di setiap *peer,* hal ini tentunya sangat memakan waktu dan rentan terjadinya inkonsistensi data.

Sedangkan kerugian dari penerapan arsitektur *clien server* antara lain:

1. Kongesti trafik menjadi masalah utama arsitektur *clien server.* Semakin banyak jumlah *request client* secara simultan, dapat mengakibatkan *server* dan jaringan kelebihan beban.
2. Jika *server* mengalami kegagalan, maka *request* dari *clien* tidak dapat dipenuhi, hal ini berlaku untuk semua *clien* yang terkoneksi dengan *server.*

2.8. Java

Menurut Fikri, Rijalul,. Adam, Fuandina, Ipam,. Prakoso, Imam,.et all (2005:15) Java merupakan bahasa pemograman berorientasi objek dan bebas *platform*, dikembangkan oleh SUN Micro System dengan sejumlah keunggulan yang memungkinkan java dijadikan sebagai bahasa pengembangan enterprise.

Adapun Krakteristik Java, antara lain :

1. Sederhana (*Simple*)
2. Berorientasi Objek (*Object Oriented*)
3. Terdistribusi (*Distributed*)
4. *Interpreted*
5. *Robust*
6. Aman (*Secure*)
7. *Architecture Neutral*
8. *Portable*
9. *Performance*
10. *Multithreaded*
11. Dinamis

2.8.1. Sejarah Java

Java lahir karena ketidakpuasan seorang insinyur di SUN Micro System bernama James Gosling. Ian tidak puas dengan compiler C++ (yang ia gunakan untuk membuat *software* yang di-embed pada peralatan elektronik) keren dinilai telalu banyak menghasilkan bug, berbiaya besar, sangat bergantung terhadap *platform*. Gosling merasa perlu membuat kompiler baru sebagai solusi terhadap sejumlah kelemahan pada C++ tersebut.

Kompiler baru tersebut diberi nama dengan Oak. Kompiler ini mirip dengan C++ tetapi dengan sejumlah pengurangan fitur yang dianggap kurang dengan C++ tetapi dengan sejumlah pengurangan fitur yang dianggap kurang menguntungkan dalam pengembangan, seperti multipelinheritance, konversi tipe secara otomatis, penggunaan pointer dan managemen memori.

Pada tahun 1994, Oak diubah namanya menjadi java. Pada era ini, java divisikan sebagai bahasa yang memiliki dukungan baik terhadap web.

2.8.2. Versi Awal Java

Versi awal Java ditahun 1996 sudah merupakan versi release sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya adalah :

1. Java.lang: Peruntukan kelas elemen-elemen dasar.
2. Java.io: Peruntukan kelas *input* dan *output*, termasuk penggunaan berkas.
3. Java.util: Peruntukan kelas pelengkap seperti kelas struktur data dan kelas kelas penanggalan.
4. Java.net: Peruntukan kelas TCP/IP, yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.
5. Java.awt: Kelas dasar untuk aplikasi antarmuka dengan pengguna (GUI)
6. Java.applet: Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah web.

2.8.3. Kelebihan dan Kekurangan Java

http://hendarsuh.wordpress.com/2011/07/12/keunggulan-dan-kelemahan_java/ 23 mei 2012

1. *Multiplatform.*

   Kelebihan utama dari Java ialah dapat dijalankan di beberapa *platform* sistem operasi komputer, sesuai dengan prinsip *tulis sekali, jalankan di mana saja*. Dengan kelebihan ini pemrogram cukup menulis sebuah program Java dan dikompilasi (diubah, dari bahasa yang dimengerti manusia menjadi bahasa mesin / *bytecode*) sekali lalu hasilnya dapat dijalankan di atas beberapa platform tanpa perubahan. Kelebihan ini memungkinkan sebuah program berbasis java dikerjakan diatas operating system Linux tetapi dijalankan dengan baik di atas Microsoft Windows. Platform yang didukung sampai saat ini adalah Microsoft windows, Linux, Mac OS dan Sun Solaris. Penyebanya adalah setiap sistem operasi menggunakan programnya sendiri-sendiri (yang dapat diunduh dari situs Java) untuk meninterpretasikan *bytecode* tersebut.

2. OPP (*Object OrientedProgramming*Pemrogram Berorientasi Objek)

   Yang artinya semua aspek yang terdapat di Java adalah objek. Java merupakan salah satu bahasa pemrograman berbasis oebjek secara murni. Semua tipe data diturunkan dari kelas dasar yang disebut *Object*. Hal ini sangat memudahkan pemrogram untuk mendesain, membuat, mengembangkan dan mengalokasi kesalahan sebuah program dengan basis Java secara cepat, tepat, mudah dan terorganisir. Kelebihan ini menjadikan Java sebagai salah satu bahasa pemograman termudah,

bahkan untuk fungsi fungsi yang advance seperti komunikasi antara komputer sekalipun.

3. Perpustakaan Kelas Yang Lengkap

   Java terkenal dengan kelengkapan *library*/perpustakaan (kumpulan program program yang disertakan dalam pemrograman java) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas Java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.

4. Bergaya *C++*

   Memiliki sintaks seperti bahasa pemrograman [C++] sehingga menarik banyak pemrogram C++ untuk pindah ke Java. Saat ini pengguna Java sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke Java. Universitas-universitas di Amerika juga mulai berpindah dengan mengajarkan Java kepada murid-murid yang baru karena lebih mudah dipahami oleh murid dan dapat berguna juga bagi mereka yang bukan mengambil jurusan komputer.

Adapun Kekurangan Java sebagai berikut :

1. Tulis sekali, perbaiki di mana saja – Masih ada beberapa hal yang tidak kompatibel antara *platform* satu dengan *platform* lain. Untuk J2SE misalnya *SWT-AWT bridge* yang sampai sekarang tidak berfungsi pada Mac OS X

2. Mudah didekompilasi. Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini dimungkinkan karena koe jadi Java merupakan *bytecode* yang menyimpan banyak atribut bahasa tingkat tinggi, seperti nama-nama kelas, metode, dan tipe data. Hal yang sama juga terjadi pada Microsoft.NetPlatform Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/di*reverse-engineer*.
3. Penggunaan memori yang banyak. Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal (lebih spesifik lagi, Delphi dan Object Pascal. Biasanya ini bukan merupakan masalah bagi pihak yang menggunakan teknologi terbaru (karena trend memori terpasang makin murah.

2.9. Eclipse

http://wi01.files.wordpress.com/2011/02/pengenalan-eclipse.pdf/10 April 2012

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua *platform* (*platform-independent*).

Berikut ini adalah sifat dari Eclipse:

1. *Multi-platform*, Target sistem operasi Eclipse adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
2. *Multi-language*, Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis

bahasa pemrograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya.

3. *Multi-role*, Selain sebagai IDE untuk pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya.

2.9.1. Sejarah Eclipse

Eclipse awalnya dikembangkan oleh IBM untuk menggantikan perangkat lunak pengembangan IBM Visual Age for Java 4.0. Produk Eclipse ini diluncurkan oleh IBM pada tanggal 5 November 2001. IBM menginvestasikan US$ 40 juta untuk pengembangannya. Sejak 5 November 2001, konsorsium Eclipse Foundation mengambil alih pengembangan Eclipse lebih lanjut.

2.9.2. Arsitektur Eclipse

Sejak versi 3.0, Eclipse pada dasarnya merupakan sebuah kernel. Apa yang dapat digunakan di dalam Eclipse sebenarnya adalah fungsi dari *plug-in* yang sudah dipasang (diinstal). Ini merupakan basis dari Eclipse yang dinamakan *Rich Client Platform* (RCP).

Berikut ini adalah komponen yang membentuk RCP:

1. Core platform
2. OSGi
3. SWT (*Standard Widget Toolkit*)
4. JFace
5. Eclipse Workbench

Secara standar Eclipse selalu dilengkapi dengan JDT (*Java Development Tools*), *plug-in* yang membuat Eclipse kompatibel untuk mengembangkan program Java, dan PDE (*Plug-in Development Environment*) untuk mengembangkan *plug-in* baru. Eclipse beserta *plug-in*-nya diimplementasikan dalam bahasa pemrograman Java. Konsep Eclipse adalah IDE adalah terbuka (*open*), mudah diperluas (*extensible*) untuk apa saja, dan tidak untuk sesuatu yang spesifik. Eclipse tidak saja untuk mengembangkan program Java, tetapi juga untuk berbagai macam keperluan. Perluasan apapun cukup dengan menginstal *plug-in* yang dibutuhkan. Apabila ingin mengembangkan program C/C++ maka telah terdapat *plug-in* CDT (*C/C++ Development Tools*) yang dapat dipasang di Eclipse untuk Eclipse menjadi perangkat untuk pengembangan C/C++.

2.9.3. Versi-Versi Eclipse

Sejak tahun 2006, Eclipse Foundation mengkoordinasikan peluncuran Eclipse secara rutin dan simultan yang dikenal dengan nama *Simultaneous Release*. Setiap versi peluncuran terdiri dari Eclipse *Platform* dan juga sejumlah proyek yang terlibat dalam proyek Eclipse. Tujuan sistem ini adalah untuk menyediakan distribusi Eclipse dengan fitur-fitur dan versi yang terstandarisasi. Hal ini juga dimaksudkan untuk mempermudah *deployment* dan *maintenance* untuk sistem *enterprise*, serta untuk kenyamanan. Peluncuran simultan dijadwalkan pada bulan Juni setiap tahunnya. Berikut ini adalah versi-versi dari eclipse antara lain:

1. Eclipse 3.0
2. Eclipse 3.1
3. Callisto

4. Europa
5. Ganymede
6. Galileo
7. Helios

2.10. Basis Data

Menurut Fathansyah,Ir (2002:9) Basis data adalah sistem yang terdiri dari atas sekumpulan tabel yang saling berhubungan (Dalam sebuah basis data di sebuah sistem komputer) dan sekumpulan program (DBMS) yang menungkinkan beberapa pemakai lain untuk mengakses dan memanipulasi table-tabel tersebut.

Menurut Fatansyah (2002 : 5), Sistem basis data pada dasarnya di buat untuk tujuan Sebagai berikut:

1. Kecepatan dan kemudahan

   Dengan sistem basis data diharapkan pengolahan data dapat lebih cepat dan lebih mudah di bandingkan dengan sistem manual.

2. Efisiensi ruang penyimpanan

   Dengan basis data,efisiensi /optimalisasi penggunaan ruang penyimpan dapat dilakukan, karena kita dapat melakukan penekanan jumlah redudansi data.

3. Keakuratan

   Pemanfaatan pengkodean atau pembentukan relasi antar data bersama dengan penerapan aturan atau batasan tipe, data, domain data keunikan data dan sebagainya yang secara ketat dapat diterapkan dalam

sebuah basis data sangat berguna untuk menekan ketidak akuratan masukan ataupun penyimpanan data.

4. Ketersediaan

   Dengan pertumbuhan data yang semakin banyak,maka di perlukan tempat penyimpanan yang sangat besar, karena itu kita perlu memilah data dalam bentuk kategori-kategori tertentu sehingga data yang tidak terlalu penting dapat di hapus.

5. Kelengkapan

   Untuk mengakomodasi kebutuhan kelengkapan data yang semaki berkembang maka kita tidak hanya dapat menambah *record-record* data tetapi juga melakukan perubahan struktur dalam basis data, baik dalam bentuk penambahan objek baru (tabel)atau dengan penambahan *field-field* baru pada suatu tabel.

6. Keamanan

   Dengan sistem basis data kita dapat memproteksi semua objek yang ada dengan berbagai tingkatan keamanan.

7. Kebersamaan Pemakaian

   Pemakai basis data biasanya terdiri dari banyak pemakai, sehingga hendaknya basis data harus bisa di akses oleh banyak pemakai dalam waktu yang bersamaan.

2.10.1. Keuntungan Databese

1. Mereduksi redudansi
2. Data dapat di *share* antar aplikasi

3. Standarisasi data dapat di lakukan
4. Batasan *security* dapat diterapkan
5. Mengelola integritas (keterjaminan akurasi) data
6. Menyeimbangkan kebutuhan yang saling konflik
7. Independesi data (obyektif DBS)

Struktur sistem Database :

1. File manager : mengelola *space* dan struktur data.
2. Database manager : menyediakan antar muka dengan data fisik
3. Query processor : menterjemahkan *query* ke instruksi yang dimengerti database *manager*
4. DML precompiler :mengkonversi perintah DML menjadi calls normal.
5. DDL compiler : menkonversi perintah DDL menjadi meta data.

2.10.2. Structured Query Language (SQL)

SQL adalah Cara berinteraksi/berkomunikasi antara pemakai dengan basis data diatur dalam suatu bahasa khusus yang ditetapkan oleh perusahaan pembuat DBMS. Bahasa itu disebut Bahasa Basis Data. Bahasa Basis Data dapat dipilah ke dalam 2 bentuk yaitu:

1. Data *Definition Language* (DDL)

   Dengan bahasa ini kita dapat membuat tabel baru, membuat indeks, mengubah tabel dan sebagainya. Perintah DDL dalam SQL meliputi :

   a. CREATE TABLE – membuat tabel baru dalam database
   b. ALTER TABLE – mengubah definisi tabel

c. DROP TABLE – menghapus tabel

d. CREATE INDEX – membuat index (search key)

e. DROP INDEX - menghapus index

2. Data *Manipulation Language* (DML)

Merupakan bentuk bahasa basis data yang berguna untuk melakukan manipulasi dan pengambilan data pada suatu basis data. Manipulasi data dapat berupa:

a. Penyisipan /Penambahan data baru ke suatu basis data

b. Penghapusan data dari suatu basis data

c. Pengubahan data di suatu basis data

Perintah SQl untuk DML (Data *Manipulation Language*) antara lain :

a. SELECT – Mengekstrak data dari tabel-tabel di database

b. UPDATE – Mengupdate data dalam tabel di database

c. DELETE – Menghapus data dari tabel di database

d. INSERT INTO – Menyisipkan/menambah data baru ke dalam tabel di database

2.10.3. MySQL

http://id.wikipedia.org/wiki/MySQL/12 April 2012

MySQL adalah salah satu jenis *database server* yang sangat terkenal. Kepopulerannya disebabkan MySQL menggunakan SQL *(Structured QueryLanguag*e) sebagai bahasa dasar untuk mengakses *database*-nya. Selain itu *software* ini bersifat *open source* sehingga tidak perlu membayar untuk menggunakannya. MySQL termasuk jenis RDBMS *(Relational Database*

*Management Syste*m). Itulah sebabnya istilah seperti tabel, baris dan kolom digunakan pada MySQL. Pada MySQL, sebuah *database* mengandung satu atau sejumlah tabel. Tabel terdiri dari sejumlah baris dan setiap baris mengandung satu atau beberapa kolom.