BAB II

LANDASAN TEORI

2.1 Konsep Dasar Sistem

Pada dasarnya sistem adalah sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel yang terorganisir, saling berinteraksi, saling tergantung satu sama lain, dan terpadu.

2.1.1 Pengertian Dasar Sistem

Definisi sistem dari pendapat beberapa pakar antara lain :

Menurut Tata Sutabri (2007: 2) sistem merupakan suatu kumpulan atau himpunan dari unsur-unsur komponen atau variabel yang terorganisir, saling berintegrasi, saling berinteraksi, saling tergantung satu sama lain dan terpadu.

Menurut Indrajit (2001: 2) mengemukakan bahwa sistem mengandung arti kumpulan-kumpulan dari komponen-komponen yang dimiliki unsur keterkaitan antara satu dengan lainnya.

Dari definisi sistem menurut beberapa para ahli tersebut dapat disimpulkan bahwa sistem adalah kumpulan dari komponen-komponen yang saling berhubungan untuk mencapai suatu tujuan.

2.1.2 Elemen Sistem

Suatu sistem, dapat terbentuk dari berbagai macam elemen, elemen-elemen tersebut adalah:

a. Masukan (input)

   Masukan atau input adalah Sekumpulan data yang akan kita olah menjadi sebuah informasi yang nantinya akan kita sajikan bagi masyarakat.

b. Keluaran (Output)

   Keluaran atau output adalah informasi-informasi yang dapat dengan mudah di peroleh, dimengerti dan dimanfaatkan oleh masyarakat.

c. Proses

   Proses adalah suatu kegiatan dimana kita mengolah seluruh data yang ada untuk menghasilkan suatu informasi .

d. Mekanisme Pengendalian

   Mekanisme Pengendalian adalah pengawasan jalannya seluruh kegiatan operasional sistem mulai dari input, proses sampai keluarannya untuk menghindari terjadinya kesalahan-kesalahan.

e. Umpan Balik

   Umpan balik adalah Masukan atau Pendapat yang diberikan oleh masyarakat mengenai sistem informasi yang kita buat sehingga kita dapat memperbaiki dan meningkatkan kinerja dari sistem informasi yang kita buat tersebut.

2.1.3 Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat-sifat tertentu, yaitu mempunyai komponen - komponen (*components*), batas (*boundary*), lingkungan luar sistem (*environments*), penghubung (*interface*), masukan (*input*), keluaran (*output*), pengolah (*process*) , dan sasaran (*objectives*) atau tujuan (*goal*).

Berikut adalah penjelasan dari karakteristik sistem :

a. Komponen Sistem

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerjasama membentuk suatu kesatuan. Komponen-komponen sistem atau elemen- lemen sistem dapat berupa subsistem atau bagian-bagian dari sistem.

Setiap sistem tidak perduli betapapun kecilnya, selalu mengandung komponen-komponen atau subsistem-subsistem. Setiap subsistem mempunyai sifat-sifat dari subsistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai suatu sistem yang lebih besar disebut dengan supra sistem, misalnya suatu perusahaan dapat disebut sebagai suatu sistem sedang industri yang merupakan sistem yang lebih besar dapat disebut dengan supra sistem. Kalau dipandang industri sebagai suatu sistem, maka perusahaan dapat disebut sebagai subsistem.

b. Batas sistem

Batas sistem (*boundary*) merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipasang sebagai suatu kesatuan. Batas suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

c. Lingkungan luar sistem

Lingkungan luar (*environment*) dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat bersifat merugikan sistem tersebut. Lingkungan luar yang menguntungkan merupakan energi dari sistem dan dengan demikian harus tetap dijaga dan dipelihara. Sedang lingkungan luar yang merugikan harus ditahan dan dikendalikan, kalau tidak maka akan mengganggu kelangsungan hidup dari sistem.

d. Penghubung sistem

Penghubung (*interface*) merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem yang lain. Keluaran (*output*) dari satu subsistem akan menjadi satu masukan (*input*) bagi subsistem yang lain dan akan melalui penghubung. Dengan penghubung satu subsistem dapat berintegrasi dengan subsistem yang lainnya membentuk satu kesatuan.

e. Masukan sistem

Masukan *(input*) adalah energi yang dimasukkan kedalam sistem. Masukan dapat berupa masukan peralatan (*maintenence input*) dan masukan sinyal (*signal input*). *Maintenance input* adalah energi yang diproses agar didapatkan keluaran. Sebagai contoh didalam sistem komputer, program adalah *maintenance input* yang digunakan untuk

mengoperasikan komputernya sedangkan data adalah signal input untuk diolah menjadi informasi.

f. Keluaran sistem

Suatu sistem dapat mempunyai suatu bagian pengolah yang akan merubah masukan menjadi keluaran. Suatu sistem produksi akan mengolah masukan berupa bahan baku dan bahan-bahan yang lain menjadi keluaran berupa barang jadi. Sistem akuntansi akan mengolah transaksi menjadi laporan keuangan dan laporan-laporan lain yang dibutuhkan oleh menejemen.

g. Sasaran sistem

Suatu sistem pasti mempunyai tujuan (*goal*) atau sasaran (objektif). Jika sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan sekali, masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem.

2.1.4 Klasifikasi Sistem

Sistem dapat diklasifikasikan dalam beberapa sudut pandangan, diantaranya sebagai berikut:

1. Sistem abstrak (*abstract system*) dan sistem fisik (*physical system*).

   Sistem abstrak adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik sedangkan sistem fisik merupakan sistem yang ada secara fisik.

2. Sistem alamiah (natural system) dan sistem buatan manusia *(human made system).*

Sistem alamiah adalah sistem yang terjadi melalui proses alam, tidak dibuat oleh manusia sedangkan sistem buatan manusia merupakan sistem yang melibatkan manusia dengan mesin.

3. Sistem deterministik *(deterministic system)* dan sistem probabilistik *(probabilistic system).*

   Sistem deterministik adalah sistem yang beroperasi dengan tingkah laku yang dapat diprediksi sedangkan sistem yang bersifat probalistik adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung probalistik.

4. Sistem terbuka (*open system*) dan sistem tertutup (*closed system*).

   Sistem terbuka adalah sistem yang berhubungan dan dipengaruhi oleh lingkungan luarnya sedangkan sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh oleh lingkungan luar lainnya.

2.2 Pengertian Aplikasi

Ada banyak pengertian mengenai aplikasi menurut beberapa ahli yaitu:

1) Aplikasi adalah satu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas (Buyens, 2001).
2) Menurut Dhanta (2009:32), aplikasi (*application*) adalah *software* yang dibuat oleh suatu perusahaan komputer untuk mengerjakan tugas-tugas tertentu, misalnya *Microsoft Word, Microsoft Excel.*

Beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu paket atau suite aplikasi *(application suite).* Contohnya adalah *Microsoft Office* dan *Open Office.org,* yang menggabungkan suatu aplikasi

pengolah kata, lembar kerja, serta beberapa aplikasi lainnya. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi. Sering kali, mereka memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna. Contohnya, suatu lembar kerja dapat dibenamkan dalam suatu dokumen pengolah kata walaupun dibuat pada aplikasi lembar kerja yang terpisah.

2.2.1 Klasifikasi Aplikasi

Aplikasi dapat digolongkan menjadi beberapa kelas, antara lain:

a) Perangkat lunak perusahaan *(enterprise)*
b) Perangkat lunak infrastruktur perusahaan
c) Perangkat lunak informasi kerja
d) Perangkat lunak media dan hiburan
e) Perangkat lunak pendidikan
f) Perangkat lunak pengembangan media
g) Perangkat lunak rekayasa produk

Pada pengertian umumnya, aplikasi adalah alat terapan yang difungsikan secara khusus dan terpadu sesuai kemampuan yang dimilikinya.

2.3 Pengertian Sistem Aplikasi

Sistem Aplikasi adalah seperangkat bagian – bagian yang saling berhubungan yang penerapannya berasal dari rancangan sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu untuk mencapai suatu hasil yang diinginkan secara efesien.

2.4 Pengertian Penerimaan Barang

Pengertian penerimaan barang menurut Ir. Kristanto Harianto dalam bukunya yang berjudul Konsep dan Perancangan Database, yaitu:

Penerimaan merupakan suatu kegiatan usaha dalam penyelenggaraan, pengurusan, dan pengaturan kiriman dalam tempat penyimpanan atau di gudang sampai tiba waktunya untuk disalurkan.

Penerimaan barang merupakan segala awal arus barang yang bergerak di gudang. Penerimaan barang dari pemasok atau rekanan memang kelihatan mudah, namun bila hal ini tidak memiliki sistem yang mengatur, maka bisa dipastikan akan mengganggu produktifitas.

Berikut adalah hal-hal penting dalam penerimaan barang :

1. Bukti pesanan barang dari gudang (untuk memastikan pesanan barang dalam spesifikasi tepat).
2. Bukti tanda barang diterima (untuk penagihan).
3. Cek bukti pemesanan dengan fisik barang.
4. Cek expired date dan kondisi barang.
5. Memasukkan barang ke penyimpanan

2.5 Teknologi *Smartphone*

Smartphone (ponsel pintar) merupakan sebuah *high-endmobile phone* di mana memiliki beberapa fungsi seperti *personal digital assistant*, alat telepon, multimedia, dan aplikasi komputer *mobile* yang telah dikombinasikan menjadi satu. *Smartphone* bisa menjalakan aplikasi yang memiliki kompleksitas yang lebih tinggi daripada *phone* biasa. *Smartphone* menyediakan sebuah *host* di mana akan

memudahkan para *user* ketika ingin melakukan *update software* dan data *plan* yang sudah disediakan untuk beberapa pengguna tertentu. Secara lebih luas ini disebut sebagai ekosistem dari sistem development sebuah *smartphone* (M. Wargo, John, 2010).

2.6 Sejarah Android

Android adalah sebuah *platform* pertama yang betul-betul terbuka dalam pengembangannya dan komprehensif untuk perangkat *mobile*, semua perangkat lunak yang ada difungsikan menjalankan sebuah *device mobile* tanpa memikirkan kendala kepemilikan yang menghambat inovasi pada teknologi *mobile* (Meier, 2009). Dalam definisi lain, android merupakan subset perangkat lunak untuk perangkat *mobile* yang meliputi sistem operasi, *middleware*, dan aplikasi inti yang dirilis oleh *Google*.

Sedangkan *Android SDK (Software Development Kit)* menyediakan *Tools* dan *API* yang diperlukan untuk mengembangkan aplikasi pada *platform Android* dengan menggunakan bahasa pemrograman *Java*. *Android* dikembangkan bersama antara *Google, HTC, Intel, Motorola, Quallcomm, T-Mobile, NVIDIA* yang tergabung dalam *OHA (Open Handset Alliance)* dengan tujuan membuat sebuah standar terbuka untuk perangkat bergerak (*Mobile Device*) (Mulyadi, 2010).

Pada tahun 2005 *Google* mengakuisisi *Android.Inc* yang pada saat itu dimotori oleh Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Yang kemudian pada tahun itu juga memulai membangun platform *Android* secara intensif. Kemudian pada tanggal 12 November 2007 *Google* bersama *Open*

Handset Alliance yaitu sebuah konsorsium perangkat mobile terbuka, merilis *Google Android SDK*, setelah mengumumkannya seminggu sebelumnya. Dan sambutannya sangat luar biasa, hampir semua media berita tentang *IT* dan *Programming* memberitakan tentang dirilisnya *Android SDK (Software Development Kit)* ini.

Ini dikarenakan Ide dari *platform Android* ini sangat menarik untuk *developer* dan programmer diseluruh penjuru dunia. Disisi lain, produsen pembuat *handset* ponsel juga segera berlomba-lomba membuat *handset* dengan *platform Android* ini.

Google bersama dengan *OHA* merilis paket *software SDK* yang lengkap untuk mengembangkan aplikasi pada perangkat *mobile*. Yaitu :
Sistem Operasi, *Middleware* dan Aplikasi Utama untuk perangkat *mobile*. Sebagai *programmer* atau *developer* bisa melakukan segalanya, mulai dari membuat aplikasi pengiriman *SMS* hanya dengan dua baris kode, hingga mengganti *event* pada *Home Screen* pearangkat *Android*. Selain itu, bahkan dengan mudah kita bisa membuat dan merubah Sistem Operasiya atau mengganti semua aplikasi default dari *Google* (Mulyadi, 2010).

2.6.1 Fitur Android

Fitur-fitur yang tersedia pada *platform Android* saat ini diantara lain :

1. *Framework* Aplikasi

   Mendukung penggantian komponen dan *reusable component*

2. *Dalvik Virtual Machine*

   Android berjalan diatas *Linux Kernel* dan dioptimalkan untuk perangkat *mobile*

3. *Integrated Browser*

   Dikembangakan berdasarkan *open source engine WebKit*

4. Grafis

   Dioptimalkan dan didukung oleh *library* grafis *2D* yang terkostumisasi, grafis *3D* berdasarkan spesifikasi *openGL ES 1.0* (Opsional akselerasi *hardware*)

5. *SQLite*

   Media untuk penyimpanan data berupa *database*

6. *Media Support*

   Mendukung *audio, video*, dan gambar (*MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF*)

7. *GSM Telephony*

8. *Bluetooth, EDGE, 3G, dan WiFi*

9. Dukungan Perangkat Tambahan

   Android dapat memanfaatkan kamera, layar sentuh, *accelerometers*, *magnetometers*, *GPS*, akselerasi *2D* (dengan perangkat orientasi, *scaling*, konversi format *pixel*) dan akselerasi grafis *3D*

10. *Multi Touch*

    Kemampuan layaknya *handset modern* yang dapat menggunakan dua jari atau lebih untuk berinteraksi dengan perangkat

11. Lingkungan *Development*

Android lengkap dan kaya termasuk perangkat *emulator*, *tools* untuk *debugging*, dan kinerja memori dan *plugin* untuk *IDE*

12. Market

Seperti kebanyakan *handphone* yang memiliki tempat penjualan aplikasi, *market* pada *android* merupakan katalog aplikasi yang dapat diunduh dan diinstal pada *handphone* melalui *internet.* (Mulyadi, 2010)

2.6.2 Arsitektur Android

1. Linux *Kernel*

   Android bukan *linux* akan tetapi *android* dibangun diatas *Linux Kernel* yatitu versi 2.6 sehingga kehandalannya bisa dipercaya. Untuk inti sistem *service linux* yang digunakan sperti keamanan, manajemen memori, proses manajemen, *network* dan *driver* model. Seperti yang terlihat digambar, *linux kernel* menyediakan *Driver* untuk Layar, Kamera, *Keypad, Wifi, Flash Memory, Audio*, dan *IPC (Interprocess Communication)* untuk mengatur aplikasi dan keamanan. *Kernel* juga bertindak sebagai lapisan abstrak antara *hardware* dan *software stacknya*.

2. *Libraries*

   Android menyertakan *library C/C++* yang digunakan oleh berbagai komponen dari sistem *Android*. Kemampuan ini disediakan kepada *developer* aplikasi melalui *framework* aplikasi *android*. Beberapa inti *library* tercantum di bawah ini:

a. *System C Library*

   Variasi dari implementasi *BSD*. Berasal pelaksanaan sistem standar *C library (libc)*, sesuai untuk perangkatan *embedded* berbasis *linux*.

b. *Media Libraries*

   Paket video berdasarkan *Open CORE*. *Library* ini mendukung pemutaran rekaman dan populer ke banyak format audio dan video, serta file gambar, termasuk *MPEG4*, *H.264*, *MP3*, *AAC,AMR*, *JPG* dan *PNG.*

c. *Surface Manager*

   Mengelola akses ke subsistem layar, lapisan komposit *2D* dan grafis *3D* dari beberapa aplikasi.

d. *Library Web Core*

   Mesin *web* modern yang *powerfull* yang baik untuk *browser Android* dengan *embedded web.*

e. *SGL*

   Untuk mesin grafis *2D*

f. *3D Libraries*

   Library 3D berdasarkan *openGL ES 1.0 API*, *Library* baik untuk digunakan pada perangkat keras dengan akselerasi *3D* atau yang disertakan, sangat optimal pada *3D software rasterizer*.

g. *Free Type*

   Bitmap dan vektor *font rendering*

h. SQLite

Mesin *database* yang kuat dan ringan dan penghubung tersedia untuk semua aplikasi.

2.6.3 Android *Runtime*

Android terdiri dari satu set *core libraries* yang menyediakan sebagian besar fungsi yang sama dengan yang terdapat pada *core library* bahasa pemrograman *Java*. Setiap aplikasi menjalankan prosesnya sendiri dalam *Android*, dengan masing-masing instan dari mesing *Dalvik Virtual Machine*. *Dalvik* dirancang agar perangkat dapat menjalankan *multiple Virtual Machine* secara efisien. Mesin *Virtual Dalvik* mengeksekusi file dalam *Dalvik Executable (.dex)*, sebuah format yang dioptimalkan untuk memori yang kecil. *Dalvik VM* berbasis, berjalan dan dikompilasi oleh kompiler bahasa *Java* yang telah ditransformasi ke dalam *.dex* format oleh tool *"dx"* yang telah disertakan olehn *Dalvik VM* bergantung pada *kernel Linux* untuk berfungsi, seperti *threading* dan manajemen memori tingkat rendahanya.

1. *Framework* Aplikasi

   Pengembang memiliki akses penuh menuju *API framework* yang sama dengan yang digunakan oleh aplikasi inti. Arsitektur aplikasi dirancang agar komponen dapat digunakan kembali (*reuse*) dengan mudah. Setiap aplikasi dapat memanfaatkan kemampuan inti (sesuai dengan batasan kemampuan yang didefinisikan oleh *framework*). Mekanisme yang sama memungkinkan komponen untuk diganti oleh

pengguna. Semua aplikasi merupakan rangakaian set layanan dan sistem termasuk :

a. Views

Komponen *user interface* yang dapat digunakan untuk membangun aplikasi termasuk *list, grids,* kotak teks, tombol dan bahkan sebuah *embedded web.*

b. Content Provider

Memungkinkan aplikasi untuk mengakses data (seperti dari daftar kontak) atau membagi data tersebut.

c. Resources Manager

Menyediakan akses ke kode non sumber lokal seperti *string*, gambar, dan tata letak *file.*

d. Notification Manager

Memungkinkan semua aplikasi menampilkan alert yang bisa dikostumisasi di dalam status bar.

e. Activity Manager

Mengelola siklus hidup aplikasi dan menyediakan navigasi umum *backstack.*

f. Aplikasi

Android telah menyertakan aplikasi inti seperti *email client, SMS,* kalender, peta, *browser*, kontak, dan lain-lain. Semua aplikasi tersebut ditulis dengan menggunakan bahasa pemrograman *Java*. Pada layer inilah *developer* atau kita menempatkan aplikasi yang dibuat. Yang

istimewa adalah pada *android* semua aplikasi baik aplikasi inti (*native*) maupun aplikasi pihak ketiga berjalan pada *layer* aplikasi dengan menggunakan *library API* yang sama. Ini berarti semua aplikasi yang dibuat untuk *android* akan memiliki akses yang setara dalam mengakses seluruh kemampuan *handset*, tanpa membedakan apakah itu merupakan aplikasi inti atau pihak ketiga. Dalam kata lain dengan *platform android* ini, *programmer* atau *developer* secara penuh akan bias mengkostumisasi perangkat *android*nya. (Mulyadi, 2010)

2.6.4 Aplikasi Android

Aplikasi *Android* ditulis dalam bahasa pemrograman *Java*, yaitu kode *Java* yang terkompilasi bersama-sama dengan data dan *file-file* sumber yang dibutuhkan oleh aplikasi yang digabungkan oleh *aapt tools* menjadi paket aplikasi *android*, sebuah *file* yang ditandai dengan akhiran *.apk file* inilah yang didistribusikan sebagai aplikasi dan diinstal pada *handset android*. *File* ini diunduh oleh pengguna ke perangkat *mobile* mereka. Semua kode dijadikan satu file *.apk,* dan kemudian kita sebut sebagai sebuah aplikasi.

Dalam banyak cara, masing-masing aplikasi *android* aktif dalam lingkungannya sendiri :

1. Secara *default* semua aplikasi berjalan sendiri dalam proses *Linux*, *android* memulai proses ketika salah satu dari kode aplikasi harus dijalankan, dan menutup proses saat tidak lagi diperlukan dan sumber daya sistem tersedia untuk aplikasi lain.

2. Setiap proses memiliki *virtual machine (VM)* sendiri, maka kode aplikasi yang dijalankan di isolasi dari semua kode aplikasi lain.
3. Secara *default*, setiap aplikasi diberikan sebuah *user ID Linux* yang unik. Perizinan (*Permissions*) diatur agar aplikasi dari *file* tersebut terlihat hanya untuk *user* yang memiliki izin untuk aplikasi itu sendiri.

Sangat memungkinkan mengatur dua aplikasi yang sama untuk berbagi *user ID* dalam hal ini masing-masing mereka akan dapat melihat file mereka satu sama lain. Untuk menjaga sumber daya sistem, aplikasi dengan *ID* yang sama juga dapat diatur untuk menjalankan proses *Linux* yang sama dan berbagi *Virtual Machine* yang sama.

2.6.5 Komponen Aplikasi

Salah satu fitur utama dari *android* adalah satu aplikasi dapat menggunakan elemen dari aplikasi lain (yang disediakan berdasarkan izin aplikasi itu). Sistem aplikasi bekerja ketika salah satu bagian aplikasi harus dijalankan dengan menginisiasi objek *Java* pada bagian tersebut. Oleh karna itu, tidak seperti kebanyakan aplikasi lainnya, pada sistem aplikasi *android* tidak ada satu *entry point* bagi semua yang ada dalam aplikasi (Misalnya : tidak ada fungsi *main()*). Sebaliknya, mereka mempunya komponen penting dalam sistem yang dapat diinisiasi dan dijalankan sesuai dengan kebutuhan. Ada empat jenis komponen, yaitu :

1. *Activity*

   Sebuah *Activity* normalnya akan menyajikan *User Interface* bagi penggunannya, dimana pengguna kemudian bisa melakukan interaksi

padanya. Misalnya, sebuah *activity* dapat memberikan daftar menu kepada pengguna dan kemudian pengguna dapat memilih atau menampilkan foto. Sebuah aplikasi pesan teks mungkin memiliki satu *activity* kedua untuk menulis pesan pada kontak yang dipilih, dan *activity* lainnya untuk memeriksa pesan-pesan lama atau mengubah pengaturan. Walaupun mereka terlihat sebagai sebuah aplikasi, tetapi masing-masing *activity* tersebut adalah mandiri. Masing-masing diimplementasikan sebagai subkelas (*subclass*) dari kelas induk (*base class*) yang bernama *Activity*.

Aplikasi mungkin terdiri hanya dari satu *activity*, tetapi ada banyak aplikasi yang terdiri hanya dari satu *activity*, tetapi ada banyak aplikasi yang terdiri dari beberapa *activity* seperti aplikasi pengiriman pesan yang sudah disebutkan diatas. *Activity* apa saja dan berapa banyak mereka, tentu saja tergantung pada aplikasi dan desain. Biasanya, satu aplikasi akan di pakai sebagai satu *UI* yang pertama diperlihatkan kepada pengguna saat aplikasi diaktifkan. Untuk pindah dari *activity* ke *activity* yang lainnya kita dapatmelakukan satu *event* misalnya mengklik tombol, memilih opsi tertentu atau melalui *trigger* tertentu.

Setiap *activity* diberikan default *window* untuk sebagai sebuah *UI*. Biasanya, *window* akan mengisi seluruh layar, tetapi mungkin lebih kecil daripada layar dan berada di atas *window* lain. Sebuah *activity* juga dapat menggunakan jendela tambahan. Misalnya, *pop-up dialog* yang meminta tanggapan pengguna di tengah-tengah pengguna informasi penting ketika mereka memilih item tertentu pada layar.

Isi (*content*) visual dari *window* disediakan oleh hirarki *view* – objek yang berasal dari kelas induk *view*. Setiap *view* mengkontrol setiap ruang didalam *window* (area persegi empat). *Parent view* berisi dan mengatur *child layout*, daun *view* (yang berada dibagian bawah hirarki) di tampilkan di area persegi empat, yang mengkontrol dan merespon terhadap aksi pengguna pada area tersebut. Misalnya, *view* menampilkan sebuah gambaran kecil dan menginisiasi sebuah aksi (*action*) ketika pengguna menyentuh gambar tersebut. *Android* memiliki sejumlah *viewm* siap pake (*ready-made*) yang dapat kita pakai tombol, *text field*, *scrolls bar*, *menu item*, *check boxes* dan yang lainya. Hirarki *view* ditempaykan dalam sebuah *window activity* oleh method *activity.setContentView()*. *Content View* adalah objek yang berada pada *root* hirarki.

2. *Services*

Services tidak memiliki visual *interface* (*UI*), namun berjalan di latar belakang (secara *background*) untuk jangka waktu tak terbatas. Misalnya, *service* mungkin mengambil data melalui jaringan atau menghitung sesuatu dan memberikan hasilkan untuk a*ctivity* yang membutuhkannya. Setiap *service* meng-extends kelas induk *service.*

3. *Broadcast Receiver*

Broadcast receiver merupakan komponen yang tidak melakukan apapun selain menerima dan bereaksi untuk menyiarkan notifikasi (*announcements*). Banyak *broadcast* berasal dalam sistem kode – misalnya, notifikasi zona waktu yang telah berubah, bahwa baterai *low*,

gambar telah diambil oleh kamera, atau pengguna mengubah tampilan bahasa. Aplikasi juga dapat menginisiasi *broadcast.* Misalnya, memberikan informasi pada aplikasi lain bahwa ada data yang telah didownload ke perangkat dan tersedia bagi mereka untuk digunakan.

4. *Content Provider*

Content Provider membuat set aplikasi data spesifikasi, tersedia untuk aplikasi lain. Data dapat disimpan dalam *file* sistem, dalam sebuah *databases SQLite*, atau cara lain yang bisa diterima. *Content Provider* meng-*extends* kelas dasar *Content Provider* untuk menerapkan metode standar yang memungkinkan aplikasi lain untuk mengambil dan menyimpan data dari jenis yang dikontrolnya. Namun, aplikasi ini tidak memanggil metode lansung. Sebaliknya mereka menggunakan Objek *Content Resolver* dan memanggil metode tersebut. *Content Resolver* dapat berkomunikasi dengan setiap *content provider*, yang bekerja sama dengan provider untuk mengelola setiap komunikasi *interprocess* yang terlibat.

Content provider menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu *Activity* (Aplikasi), misalnya ketika kita menggunakan suatu aplikasi berbasis peta (*Map*), dan navigasi, maka disinilah *content provider* akan berperan. Setiap kali ada permintaan yang harus di tangani oleh komponen tertentu, a*ndroid* akan memastikan bahwa proses aplikasi dari komponen berjalan, memulainya jika diperlukan, dan mengumumkan ketersediaan objek instant dari komponen, juga menciptkan objek instant jika diperlukan.

5. Versi Sistem Operasi *Android*

1. Versi 1.0 dirilis 23 September 2008
2. Versi 1.1 dirilis 9 February 2009
3. Versi 1.5 (*Cupcake*)
4. Versi 1.6 (*Donut*)
5. Versi 2.0 / 2.1 (*Eclair*)
6. Versi 2.2 (*Froyo*)
7. Versi 2.3 (*Gingerbread*) yaitu versi yang digunakan saat membangun aplikasi ini.
8. Versi 3.0 / 3.1 / 3.2 (*Honeycomb*)
9. Versi 4.0 (*IceCreamSandwich*)
10. Versi 4.1 / 4.2 (*Jelly Bean*)

2.7 Internet

Internet adalah sebuah solusi jaringan yang dapat menghubungkan beberapa jaringan local yang ada pada suatu daerah, kota, bahkan pada sebuah negara. Dengan adanya internet kita dapat menghubungkan beberapa jaringan local yang ada pada setiap tempat. Tidak hanya itu, juga dengan adanya internet setiap orang dapat melakukan kegiatan apa saja, mulai dari sekedar mengirim informasi secara internal, atau bahkan melakukan pembelanjaan pada *took on-line* antar negara.

Internet memasuki daerah era baru dengan digunakan WWW atau yang sering disebut ***World Wide Web*** (Jaringan Dunia Luas) yaitu sebuah bagian dari internet yang sangat dikenal dalam dunia internet, sehingga memungkinkan setiap

orang atau organisasi yang mempunyai alamat di internet dapat berhubungan dengan komputer lain dengan tidak memandang lokasi, jenis komputer, serta system operasi yang digunakan, dengan adanya WWW seorang pengguna dapat menampilkan halaman virtual yang disebut dengan website. (Fathansyah, 2002)

2.7.1 World Wide Web (WWW)

World Wide Web (WWW) atau bisa disebut dengan Web merupakan salah satu sumber daya internet yang berkembang pesat. Saat ini, informasi web di distribusikan melalui pendekatan *hyperlink*, yang memungkinkan suatu teks, gambar, ataupun objek yang lain menjadi acuan untuk membuka halama-halaman web yang lain.

Berkaitan dengan web, terdapat beberapa istilah penting seperti situs web (*web site)*, *homepage*, halaman web (*web page*), dan URL.

2.8 Perangkat Lunak Pendukung

2.8.1 Eclipse

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*:

a. *Multi-platform* : Target sistem operasi *Eclipse* adalah *Microsoft Windows*, *Linux*, *Solaris*, *AIX*, *HP-UX* dan *Mac OS X*.

b. *Mulit-language* : *Eclips*e 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.

c. *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.

Eclipse pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari *Eclipse* yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan *plug-in*.

2.8.2 NetBeans IDE

NetBeans adalah *Integrated Development Environment (IDE)* berbasiskan Java dari *Sun Microsistems* yang berjalan di atas *Swing. Swing* sebuah teknologi Java untuk pengembangan aplikasi Desktop yang dapat bejalan di berbagai macam *platforms* seperti *Windows, Linux, Mac OS X* dan *Solaris*.

Suatu IDE adalah lingkup pemrograman yang diintegrasikan kedalam suatu aplikasi perangkat lunak yang menyediakan pembangun *Graphic User Interface (GUI),* suatu text atau kode editor, suatu *compiler* atau *interpreter* dan suatu *debugger. Netbeans* merupakan *software development* yang *Open Source*, dengan kata lain *software* ini di bawah pengembangan bersama, bebas biaya. *NetBeans* merupakan sebuah proyek kode terbuka yang sukses dengan pengguna yang sangat luas, komunitas yang terus tumbuh, dan memiliki hampir 100 mitra.

Sun Microsistems mendirikan proyek kode terbuka NetBeans pada bulan Juni 2000 dan terus menjadi sponsor utama. Saat ini terdapat dua produk : *NetBeans IDE* dan *NetBeans Platform.*

a. *NetBeans IDE* adalah sebuah lingkungan pengembangan sebuah kakas untuk pemrogram menulis, mengompilasi, mencari kesalahan dan menyebarkan program. *Netbeans IDE* ditulis dalam Java namun dapat mendukung bahasa pemrograman lain. Terdapat banyak modul untuk memperluas *Netbeans IDE. Netbeans IDE* adalah sebuah produk bebas dengan tanpa batasan bagaimana digunakan.
b. *NetBeans Platform* sebuah fondasi yang modular dan dapat diperluas yang dapat digunakan sebagai perangkat lunak dasar untuk membuat aplikasi desktop yang besar. Mitra ISV menyediakan *plug-in* bernilai tambah yang dapat dengan mudah diintegrasikan ke dalam *platform* dan dapat juga digunakan untuk membuat kakas dan solusi sendiri.

Kedua produk adalah kode terbuka *(open source)* dan bebas *(free)* untuk penggunaan komersial dan non komersial. Kode sumber tersedia untuk guna ulang dengan lisensi *Common Development and Distribution License (CDDL).*

2.8.3 AMPPS

AMPPS adalah setumpuk *Apache Web Server, MySQL, PHP, Perl, Python* dan *Softaculous.* Dengan AMPPS dapat membuat beberapa database *MySQL* dan mengaturnya dengan mudah pada *phpMyAdmin.*