BAB 2
LANDASAN TEORI

2.1 Aplikasi

Aplikasi adalah penggunaan dalam suatu komputer, instruksi (*Instruction*) atau pernyataan (*Statement*) yang disusun sedemikian rupa sehingga komputer dapat memproses input menjadi output. [3]

Aplikasi merupakan perangkat lunak yang berada di dalam sistem dan digunakan untuk mengolah data untuk menjadi suatu informasi yang berguna bagi penggunanya. Aplikasi juga dapat diartikan sebagai suatu program yang siap untuk digunakan yang dibuat untuk fungsi – fungsi tertentu dengan harapan dapat membantu memudahkan penggunanya.

2.2 Data, Informasi dan Sistem Informasi

2.2.1 Data

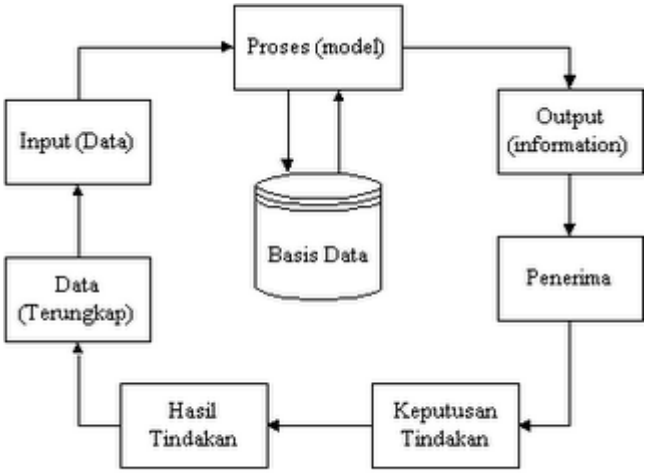
Data merupakan sesuatu yang tidak mempunyai arti bagi penggunanya dan masih memerlukan adanya suatu pengolahan terhadap data tersebut. Data bisa berbentuk suatu kejadian, benda, aktifitas, gambar, huruf dan sebagainya. Data adalah bahan mentah untuk menjadi suatu informasi.

2.2.2 Informasi

Informasi merupakan data yang telah di proses atau diolah sehingga memiliki nilai tertentu bagi penggunanya. Keberadaan data sangatlah penting karena data merupakan bahan utama untuk didapatkannya suatu informasi yang dibutuhkan.

Informasi juga diartikan sebagai hasil proses, manipulasi, dan penataan dari sekelompok data yang memiliki sebuah nilai pengetahuan bagi penggunanya.

Sumber dari suatu informasi adalah data. Data yang diolah melalui suatu model menjadi informasi, penerima kemudian menerima informasi tersebut, membuat suatu keputusan dan melakukan tindakan, yang berarti menghasilkan suatu tindakan yang lain yang akan membuat sejumlah data kembali. Data tersebut akan ditangkap sabagai input, diproses kembali lewat suatu model dan seterusnya membentuk suatu siklus. Siklus informasi ini dapat digambarkan sebagai berikut :



Gambar 2.1 Siklus Informasi

2.2.3 Sistem Informasi

Sistem informasi merupakan suatu sistem di dalam suatu organisasi yang mengintregasikan kebutuhan pengolahan transaksi harian yang mendukung sebuah operasi yang bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan laporan-laporan yang dibutuhkan untuk pihak luar yang terkait dalam suatu organisasi tersebut.

Definisi lain sistem informasi adalah kerangka kerja yang mengkoordinasikan sumber daya (manusia dan komputer) untuk mengubah masukan (input) menjadi keluaran (informasi) guna mencapai sasaran-sasaran dan tujuan perusahaan.

2.3 Sistem Operasi Android

2.3.1 Pengenalan Sistem Operasi Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel/*smartphone*. Kemudian untuk mengembangkan Android dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak dan telekomunikasi termasuk Google, HTC, Intel Motorola, Qualcomm, T-Mobile, dan Nvidia.

Pada saat perilisan perdana Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan open source pada perangkat mobile. Di lain pihak, Google merilis kode – kode Android di bawah lisensi Apache, Sebuah lisensi perangkat lunak dan open platform perangkat seluler.

Di dunia ini terdapat dua jenis distributor system operasi android. Pertama yang mendapat dukungan penuh dari Google atau Google *Mail Services* (GMS) dan kedua adalah yang benar – benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD). Sekitar September 2007 Google mengenalkan Nexus One, salah satu jenis smartphone yang menggunakan android sebagai sistem operasinya. Telepon seluler ini diproduksi oleh HTC *Corporation* dan tersedia di pasaran pada 5 Januari 2010. Pada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja android ARM Holdings, Atheros *Communications*, diproduksi oleh Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dab Vodafone Group Plc. Seiring pembentukan *Open Handset Alliance*, OHA mengumumkan produk perdana mereka, android, perangkat *mobile* yang merupakan modifikasi kernel Linux 2.6. Sejak android dirilis telah dilakukan berbagai pembaruan berupa perbaikan bus dan penambahan fitur baru. Pada masa saat ini sebagian besar *vendor – vendor smartphone* sudah memproduksi

smartphone berbasis android, vendor – vendor itu antara lain HTC, Motorola, Samsung, LG, HKC, Huawei, Archos, Webstation Camangi, Dell, Nexus, SciPhone, WayteQ, Sony Ericsson, Acer, Phillips, T-Mobile, Nexian, IMO, Asus dan masih banyak lagi *vendor smartphone* di dunia yang memproduksi android. Hal ini karena android itu adalah sistem operasi yang *open source* sehingga bebas disistribusikan dan dipakai oleh *vendor* manapun.

Tidak hanya menjadi sistem operasi di *smartphone*, saat ini android menjadi pesaing utama dari Apple pada sistem operasi Table PC. Pesatnya pertumbuhan android selain faktor yang disebutkan di atas adalah karena android itu sendiri adalah platform yang sangat lengkap baik itu sistem operasinya, aplikasi dan Tool Pengembangannya, market aplikasi android serta dukungan yang sangat tinggi dari komunitas open source di dunia, sehingga android terus berkembang pesat baik dari segi teknologi maupun dari segi jumlah *device* yang ada di dunia. [3]

2.3.2 Versi Android

Terdapat keunikan dari nama sistem operasi android yaitu digunakannya nama – nama makanan penutup (*Dessert*) dan setiap kenaikan versi nama – nama sistem operasinya memiliki huruf awalan berutan sesuai abjad. Berikut versi – versi android yang pernah dirilis :

1. Android 1.1

   Pertama kali dirilis adalah pada 9 Maret 2009. Awalnya versi sistem operasi android ini dirilis untuk perangkat T-Mobile G1 saja. Versi ini merupakan update dari untuk memperbaiki beberapa bugs dan mengganti API serta menambahkan beberapa fitur lain.

2. Android 1.5 (Cupcake)

   Versi ini pertama kali dirilis pada pertengahan Mei 2009. Dimulai dari versi ini penamaan versi android mulai menggunakan nama makanan pencuci mulut (dessert). Sistem operasi ini berbasiskan kernel Linux 2.6.27 dan menambahkan beberapa update serta UI baru dari versi android yang sebelumnya.

3. Android 1.6 (Donut)

   Dirilis pertama kali pada September 2009. Memiliki peningkatan fitur pencarian dan UI yang lebih user friendly. Pada versi ini sudah mendukung teknologi CDMA/EVDO, 802.1x, VPNs kemudian support layar dengan resolusi WVGA.

4. Android 2.0/2.1 (Éclair)

   Versi ini dirilis pertama kali pada 3 Desember 2009. Terjadi penambahan fitur untuk pengoptimalan hardware, peningkatan Google Maps 3.1.2, perubahan UI dengan browser baru dan dukungan HTML5. Beberapa versi updatenya antara android 2.0 kemudian 2.0.2 dan terakhir 2.1.

5. Android 2.2 (Froyo : Frozen Yoghurt)

   Dirilis pertama kali pada bulan Mei 2010 pada smartphone Google Nexus One. Pada versi ini sudah support terhadap Adobe Flash Player 10.1, peningkatan pada kecepatan membuka dan menutup aplikasi serta penggunaan SD Card sebagai media penyimpanan aplikasi.

6. Androidd 2.3 (Gingerbread)

   Pertama kali diperkenalkan pada Desember 2010. Dirancang untuk memaksimalkan kemampuan aplikasi dan game serta mulai digunakannya Near Field Communication (NFC). Memiliki perbaikan terhadap dukungan layar dengan resolusi WXGA dan diatasnya.

7. Android 3.0/3.1 (Honeycomb)

   Dirilis pada 22 Februari 2011 dan Motorola Xoom adalah yang pertama kali menggunakannya. Versi ini didesain khusus untuk pengoptimalan penggunaan tablet PC.

8. Android 4.0 (Ice Scream Sandwidch)

   Dirilis pada 19 Oktober 2011 dan pertama kali digunakan pada smartphone Samsung Galaxy Nexus. Secara teori semua perangkat android versi sebelumnya, Gingerbread dapat di-update ke android 4.0 Ice Scream Sandwidch.

9. Android 4.1 (Jelly Bean)

   Android Jelly Bean dibuat dengan memiliki keunggulan dan fitur baru. Penambahan baru itu diantaranya adalah meningkatkan input keyboard, desain baru fitur pencarian, UI yang baru serta pencarian melalui Voice Search yang lebih cepat. Android Jelly Bean ini pertama kali digunakan pada produk tablet Asus yaitu Google Nexus 7.

2.3.3 Arsitektur Android

Arsitektur android terdiriri dari beberapa lapisan yang mendukung fungsi–fungsi spesifik dari sistem operasi. Berikut lapisan – lapisan arsitektur android :

1. *Applications and Widgets*

   Merupakan layer dimana kita berhubungan dengan aplikasi saja. Biasanya dengan mendownload aplikasi kemudian dilakukan instalasi dan menjalankan aplikasi tersebut. Pada layer terdapat aplikasi ini seperti klien email, program SMS, kalender, peta, browser, kontak dan sebagainya dan semua aplikasi dibuat dengan bahasa pemrograman Java.

2. *Applications Frameworks*

   Android termasuk kedalam *Open Development Platform* yang menawarkan kepada pengembang untuk membangun aplikasi bagus dan inovatif. Contoh dari *service* yang berjalan di background adalah *Asyncronous Task* yang merupakan sebuah *class* abstrak yang digunakan untuk melakukan operasi pada background dan menampilkannya di UI. Pengembang juga memiliki akses menuju API *framework* seperti yang dilakukan aplikasi pada kategori inti.

   Komponen – komponen yang termasuk di dalam *Applications Frameworks* adalah sebagai berikut :

   a. *Views*
   b. *Content Provider*
   c. *Resource Manager*
   d. *Notification Manager*
   e. *Activity Manager*

3. *Libraries*

   Merupakan layer dimana fitur – fitur android berada. Android menyediakan satu set *library* – *library* dalam bahasa C/C++ yang dapat digunakan oleh berbagai komponen android. Berikut beberapa *library* yang ada di android :

   a. *Libraries* media utnuk pemutaran media gambar, audio maupun video.
   b. *Libraries* untuk manajeman tampilan
   c. *Libraries Graphics* SGL dan OpenGL untuk grafis 2D dan 3D
   d. *Libraries* SQLite untuk dukungan database
   e. *Libraries* SSL dan WebKit terintegrasi dengan web browser dan security.
   f. *Libraries* LiveWebcore mencakup modern web browser dengan engine embedded web view.
   g. *Libraries* 3D untuk implementasi OpenGL ES 1.0 API's.

4. Android *Runtime*

   Merupakan layer yang membuat aplikasi android dapat dijalankan dimana dalam prosesnya menggunakan implementasi Linux. *Dalvik Virtual Machine* (DVM) adalah mesin yang membentuk dasar kerangka aplikasi android. Terdapat dua bagian dalam android Runtime, yaitu :

   a. *Core Libraries*

      Aplikasi android dibuat menggunakan bahasa pemrograman Java. Dalvik Virtual Machine bukan merupakan Virtual Machine Java sehingga dibutuhkan sebuah libraries yang dapat menterjemahkan bahasa Java/C yang ditangan oleh bagian Core Libraries ini.

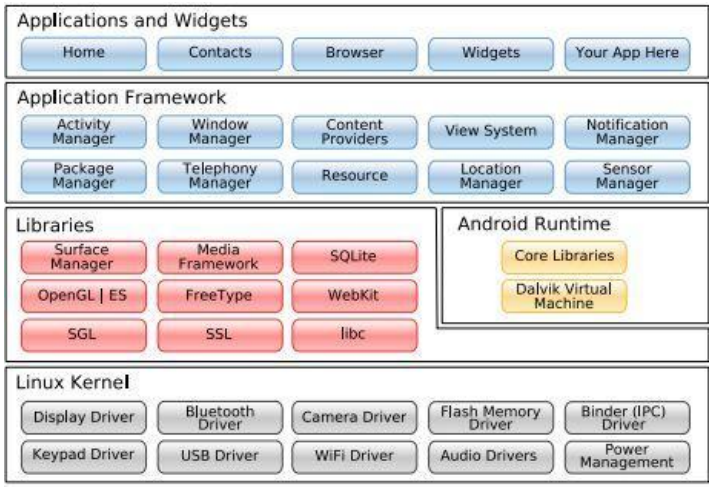
   b. *Dalvik Virtual Machine* (DVM)

      Virtual mesin berbasis register yang dioptimalkan untuk menjalankan fungsi – fungsi secara efisien, dimana merupakan pengembangan yang mampu membuat Linux kernel untuk melakukan *threading* dan manajemen tingkat rendah.

5. Linux Kernel

   Merupakan layer yang dimana inti dari system operasi android itu berada. Berisi file – file sistem yang mengatur sistem processing, memory, resources, drivers dan sistem – sistem operasi android lainnya. Linux kernel yang digunakan android adalah linux kernel release 2.6.

   Untuk lebih mudah arsitektur android dapat dilihat pada gambar 2.2:



Gambar 2.2 Arsitektur Android [7]

2.3.4 Komponen Android

Aplikasi android dibangun menggunakan bahasa pemrograman Java. Untuk membuat aplikasi android terdapat beberapa komponen utama didalamnya, yaitu :

1. *Activites*

   Merupakan suatu tampilan (*User Interface*) untuk pengguna melakukan interaksi dengan aplikasi android. Sebuah aplikasi android biasanya memiliki lebih dari satu *activity* tergantung tujuan dan desain aplikasi tersebut. Untuk berpindah dari satu activity ke *activity* lain dapat dilakukan dengan suatu event seperti klik tombola tau menggunakan trigger. Dalam hirarki *activity* dinyatakan dengan method *Activity*.setContentView().

2. *Service*

   Service tidak memiliki *Graphic User Interface* (GUI) tetapi *service* berjalan secara *background*. Sebagai contoh media player sedang memutar musik dari daftar lagu yang ada, aplikasi ini akan memiliki dua atau lebih activity yang memungkinkan pengguna untuk melakukan aktifitas lain seperti menulis sms selama musiknya masih berjalan. Untuk melakukan itu, *activity* media *player* dapat menjalankan *service* dimana *service* tersebut akan dijalankan pada thread utama dari proses aplikasi.

3. *Broadcast Receiver*

   Broadcast Receiver adalah komponen aplikasi yang menanggapi *system-wide broadcast announcements*. Beberapa contoh *broadcast* yang dikirimkan oleh sistem operasi android adalah munculnya notifikasi ketika adanya aplikasi baru yang ter-install/ter-*update*, layar mati , *booting* sistem operasi selesai, SMS atau telepon masuk, dan sebagainya. *Broadcast Receiver* juga tidak memiliki tampilan, tapi dia dapat menjalankan suatu *activity* atau menampilkan notifikasi di *Notification* Bar.

4. *Content Provider*

   Sebuah *content provider* mengatur sekumpulan data aplikasi yang terbagi (shared). Dengan content provider dapat menyimpan data di filesystem, sebuah database SQLite, di web, atau di metode penyimpanan data lainnya yang bisa diakses oleh aplikasi. Melalui *content provider*, aplikasi lain bisa memberikan *query* atau bahkan bisa memodifikasi, tentunya jika content *provider* mengijinkan aksesnya. Sebagai contoh, sistem android menyediakan *content provider* yang mengatur informasi kontak user.

5. *Intent*

   Intent merupakan pesan Asynchronous yang memungkinkan aplikasi untuk memberikan request secara fungsional dari komponen yang berbeda-beda di dalam sistem android.

2.3.5 Android SDK (Software Development Kit)

Android SDK adalah tools API (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Android merupakan *subset* perangkat lunak untuk ponsel yang meliputi sistem operasi, *middeware* dan aplikasi kunci yang di-*release* oleh Google. Saat ini disediakan Android SDK (*Software Development Kit*) sebagai alat bantu dan API untuk mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Sebagai *platform* aplikasi netral, Android memberi kesempatan untuk membuat aplikasi yang dibutuhkan yang bukan merupakan aplikasi bawaan *handphone*/*smartphone*.

2.4 Structured Analysis and Design (SAD)

2.4.1 Pengenalan Structured Analysis and Design (SAD)

Analisis terstruktur terfokus pada aliran data dan proses bisnis dan perangkat lunak. Analisis ini disebut *process oriented*. Analisis terstruktur sederhana dalam konsep. Para analisis menggambarkan serangkaian proses dalam bentuk diagram alir data (*data flow diagram*) yang menggambarkan proses yang ada atau yang di usulkan bersama-sama dengan *input*, *output* dan file mereka.

Setelah tahapan analisis selesai, maka usulan kebutuhan sistem harus diterjemahkan menjadi sistem informasi berbasis komputer. Proses mengubah usulan kebutuhan menjadi perangkat lunak bukan hal yang mudah, harus ada beberapa langkah yang digunakan untuk mempermudah dan menjamin perangkat lunak yang dihasilkan berkualitas. Langkah awal desain biasanya dimulai dengan pemodelan sistem. Model digunakan untuk menyederhanakan cara mengomunikasikan proses-proses bisnis yang harus dilakukan sistem dengan cara yang formal antar pemain pengembangan sistem.

Pemodelan proses adalah cara formal untuk menggambarkan bagaimana bisnis beroperasi. Mengilustrasikan aktivitas-aktivitas yang dilakukan dan bagaimana data berpindah di antara aktivitas-aktivitas itu. Ada banyak cara untuk

mempresentasikan proses model. Cara yang paling populer adalah dengan menggunakan *Data Flow Diagram* (DFD).

2.4.2 Data Flow Diagram (DFD)

Suatu diagram arus data atau *Data Flow Diagram* (DFD) adalah penyajian grafis dari sebuah sistem yang mempergunakan empat bentuk simbol untuk mengilustrasikan bagaimana data mengalir melalui proses-proses yang saling tersambung.

Ada empat elemen yang menyusun suatu DFD, yaitu:

1. Proses

   Aktivitas atau fungsi yang dilakukan untuk alasan bisnis yang spesifik, biasa berupa manual maupun terkomputerisasi.

2. *Data Flow*

   Satu data tunggal atau kumpulan logis suatu data, selalu diawali atau berakhir pada suatu proses.

3. *Data Store*

   Kumpulan data yang disimpan dengan cara tertentu. Data yang mengalir disimpan dalam *data store*. Aliran data di-*update* atau ditambahkan ke *data store*.

4. *External Entity*

   Orang, organisasi, atau sistem yang berada di luar sistem tetapi berinteraksi dengan sistem.

DFD terdiri dari beberapa *level*, diantaranya sebagai berikut:

1. *Context Diagram*

   Context Diagram (Diagram Konteks) merupakan DFD pertama dalam proses bisnis. Menunjukkan dimana proses bisnis berada. Menunjukkan semua proses bisnis dalam 1 proses tunggal (proses 0). *Context diagram* juga menunjukkan semua entitas luar yang menerima informasi dari atau memberikan informasi ke sistem.

2. *Level 0 Diagram*

   Menunjukkan semua proses utama yang menyusun keseluruhan sistem. Level ini juga menunjukkan komponen internal dari proses 0 dan menunjukkan bagaimana proses-proses utama direlasikan menggunakan *data flow*. Pada level ini juga ditunjukkan bagaimana proses-proses utama terhubung dengan entitas eksternal. Pada level ini juga dilakukan penambahan *data store*.

3. *Level 1 Diagram*

   Umumnya diagram level 1 diciptakan dari setiap proses utama dari level 0. Level ini menunjukkan proses-proses internal yang menyusun setiap proses-proses utama dalam level 0, sekaligus menunjukkan bagaimana informasi berpindah dari satu proses ke proses yang lainnya. Jika misalnya proses induk dipecah, katakanlah menjadi 3 proses anak, maka 3 proses anak ini secara utuh menyusun proses induk.

4. *Level 2 Diagram*

   Menunjukkan semua proses yang menyusun sebuah proses pada level 1. Bisa saja penyusunan DFD tidak mencapai level 2 ini. Atau mungkin harus dilanjutkan ke level berikutnya (level 3, level 4, dan seterusnya).

2.5 Object-Oriented Programming (OOP)

2.5.1 Pengenalan Object-Oriented Programming (OOP)

Object-Oriented Programming (OOP) adalah sebuah pendekatan untuk pengembangan atau *development* suatu *software* dimana dalam struktur *software* tersebut didasarkan kepada interaksi *object* dalam penyelesaian suatu proses atau tugas. Interaksi tersebut mengambil form dari pesan – pesan dan mengirimkannya kembali antar *object* tersebut. *Object* akan merespon pesan tersebut menjadi sebuah tindakan atau metode. Adapun konsep pada pemrograman berorientasi objek ini yaitu:

1. *Object*

   Object adalah sebuah struktur yang menggabungkan data dan prosedur untuk bekerja bersama – sama.

2. *Abstraction*

   Ketika membangun *objects* dalam aplikasi OOP, adalah penting untuk menggabungkan konsep *abstraction* ini. Jika membangun aplikasi shipping, maka harus membangun *object* produk dengan atribut seperti ukuran dan berat. Warna adalah contoh informasi yang tidak ada hubungannya dan harus dibuang. Tetapi ketika akan membangun *order-entry application*, warna menjadi penting dan harus termasuk atribut *object* produk.

3. Enkapsulasi

   Sebuah proses dimana tidak ada akses langsung ke data yang diberikan, bahkan hidden. Jika ingin mendapat data, harus berinteraksi dengan *object* yang bertanggung jawab atas data tersebut.

4. *Polymorphisms*

   Kemampuan dua buah *object* yang berbeda untuk merespon pesan permintaan yang sama dalam suatu cara yang unik. Contohnya, dapat mengirim pesan *print* ke *object* printer yang akan mencetak pada printer, dan juga dapat mengirim pesan yang sama ke *object screen* yang akan menuliskan pada *screen* monitor. Dalam OOP, diterapkan tipe *polymorphism* melalui proses yang disebut *overloading*. Dapat dilakukan dalam implementasi metode yang berbeda pada sebuah *object* yang mempunyai nama yang sama.

5. *Inheritance*

   Penggunaan *inheritance* dalam OOP untuk mengklasifikasikan *objects* dalam program sesuai karakteristik umum dan fungsinya. Hal ini akan membuat pekerjaan bersama *object* lebih mudah dan lebih intuitif. Hal ini juga membuat programming lebih mudah karena memungkinkan untuk mengkombinasikan karakteristik umum kedalam *object parent* dan mewariskan karakteristik ini ke *child object*.

2.5.2 Unified Modelling Language (UML)

Unified Modelling language (UML) adalah keluarga notasi grafis yang didukung oleh meta – model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO). Definisi ini merupakan definisi yang sederhana. Pada kenyataanya, pendapat orang – orang tentang UML berbeda satu sama lain. Hal ini dikarenakan oleh sejarahnya sendiri dan oleh perbedaan persepsi tentang apa yang membuat sebuah proses rancang-bangun perangkat lunak efektif [8].

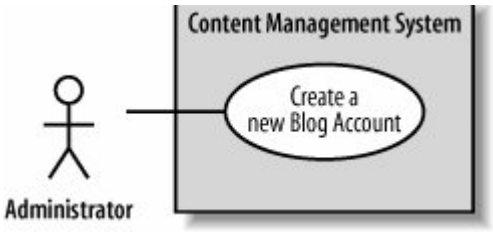
Bahasa pemrograman grafis telah ada di industri perangkat lunak sejak lama. Pemicu utaman di balik semuanya adalah bahasa pemrograman berada pada tingkat abstraksi yang tidak terlalu tinggi untuk diskusi tentang desain. UML merupakan standar yang relatif terbuka yang dikontrol oleh *Object Management Group* (OMG), sebuah konsorsium terbuka yang terdiri dari banyak perusahaan. OMG dibentuk untuk membuat standar – standar yang mendukung interoperabilitas, khususnya interoperabilitas sistem berorientasi objek. OMG mungkin lebih dikenal dengan standar – standar CORBA (*Commom Object Request Broker Architecture*). Uml lahir dari penggabungan banyak bahasa pemodelan grafis berorientasi objek yang berkembang pesat pada akhir 1980-an dan awal 1990-an.

2.5.3 Use Case Diagram

Use case adalah teknik untuk merekam persyaratan fungsional sebuah sistem. Sebuah use case juga menggambarkan cara sistem berperilaku untuk memenuhi kebutuhan. Use *case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang sistem tersebut digunakan. Dalam bahasan *Use case*, para pengguna disebut sebagai aktor. Aktor merupakan sebuah peran yang dimainkan seorang pengguna dalam kaitannya dengan sistem. Aktor tidak harus manusia, jika sebuah sistem

melakukan sebuah layanan untuk sistem komputer lain, sistem lain tersebut merupakan aktor.

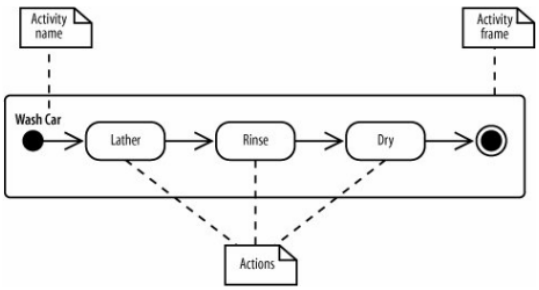
Berikut contoh bentuk *Use Case Diagram* yang mengkomunikasikan antara aktor *Use Case* di dalam sebuah sistem :



Gambar 2.3 Contoh Use Case Diagram [9]

2.5.4 Activity Diagram

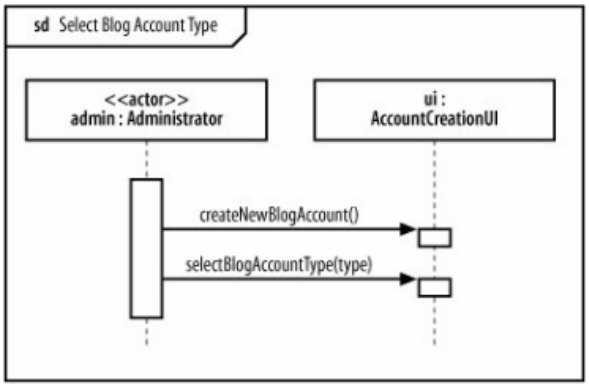
Activity diagram adalah teknik untuk menggambrakan logika prosedural, proses bisnis, dan jalur kerja. Dalam beberapa hal, diagram ini memainkan peran mirip sebuah diagram alir, tetapi perbedaan prinsip antar diagram ini dan notasi diagram alir adalah diagram ini mendukung sifat pararel. *Activity diagram* memungkinkan siapapun yang melakukan proses untuk memilih urutan dalam melakukannya. Dengan kata lain, diagram hanya menyebutkan aturan – aturan rangkaian dasar yang harus kita ikuti. Hal ini penting untuk pemodelan bisnis, karena proses – proses sering muncul secara pararel. Ini juga berguna pada algoritma yang bersamaan, di mana urutan – urutan independen dapat melakukan hal – hal secara pararel. Contoh sederhana penggunaan *activity diagram* pada kasus proses mencuci mobil dapat dilihat pada gambar 2.7 :



Gambar 2.4 Contoh Activity Diagram [9]

2.5.5 Sequence Diagram

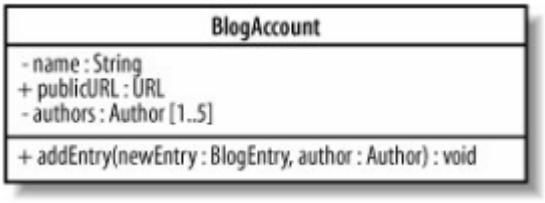
Sequence Diagram menunjukan interaksi antar diagram yang berkolaborasi dalam beberapa sifat. Sebuah *sequence* diagram secara khusus menjabarkan sifat skenario tunggal. Diagram tersebut menunjukan sejumlah objek contoh dan pesan – pesan yang melewati objek – objek ini di dalam *Use case*. Berikut contoh bentuk dari *sequence diagram* :



Gambar 2.5 Contoh *Sequence Diagram* [9]

2.5.6 Class Diagram

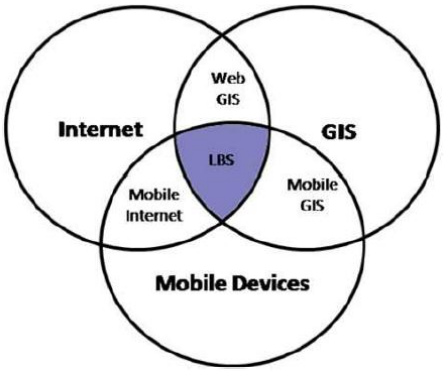
Class diagram mendeskripsikan jenis – jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat di antara mereka. *Class diagram* juga menunjukan properti dan operasi sebuah class dan batasan – batasan yang terdapat dalam hubungan – hubungan objek tersebut. UML menggunakan istilah fitur sebagai istilah umum yang meliputi properti dan operasi sebuah class.



Gambar 2.6 Contoh Class Diagram [9]

2.6 Location Based Service

Location Based Service (LBS) atau layanan berbasis lokasi adalah layanan informasi yang dapat diakses melalui *mobile device* dengan menggunakan *mobile network*, yang dilengkapi dengan kemampuan untuk memanfaatkan lokasi dari *mobile device* tersebut. LBS memungkinkan untuk melakukan komunikasi dan interaksi dua arah. LBS ini dapat digambarkan sebagai suatu layanan yang berada pada pertemuan tiga teknologi yaitu *Geographic Information System* (GIS), Internet *Service* dan *mobile device*, seperti pada gambar 2.7 : [5]



Gambar 2.7 Teknologi Location Based Service [5]

2.6.1 Metode Penentu Lokasi

LBS merupakan layanan yang mampu bereaksi aktif pada suatu perubahan posisi sehingga dapat mendeteksi letak objek dan memberikan layanan sesuai dengan letak objek yang telah diketahui tersebut. Pada teknologi LBS berbasis jaringan seluler, penentuan posisi pada sebuah peralatan komunikasi ditentukan berdasarkan posisi relatif terhadap lokasi BTS (*Base Transceiver Station*). Dalam menentukan posisi dari sebuah perangkat mobile yang sedang aktif, secara umum terdapat tiga tingkat metode yang digunakan saat ini, diantaranya adalah sebagai berikut : [5]

1. Metode *Basic Positioning* berbasis pada *Cell Identification* (Cell ID)

   Penentuan posisi didasarkan pada daerah geografis yang tercakup oleh sebuah *Cell* berhubungan dengan daerah cakupan dari sinyal radio. Ketika sebuah *handphone* terhubung secara aktif dengan sebuah *base station*,

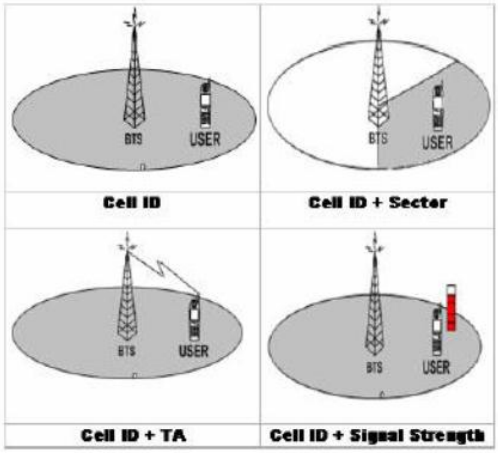
berarti *handphone* tersebut diasumsikan berada dalam *cell* dari *base station* tersebut. Untuk mengukur jarak dan arah *handset* dari *base station* tidak dapat diketahi dengan pasti, oleh karena itu untuk lebih meningkatkan lagi akurasi hasil pencarian pada metode ini seringkali dikombinasikan dengan metode lain seperti:

a. *Timing Advance* (TA)

   Dengan menggunakan TA ini metode Cell ID akan ditambahkan sebuah fungsionalitas untuk menghitung *Round Trip Time* (RTT), yaitu waktu transmisi sebuah *frame* (dari *base station* ke *handphone*) dan waktu sebuah *frame* (dari *handphone* ke *station*). Dengan tambahan ini jarak antara *handphone* dan *base station* dapat ditentukan dengan keakuratan 50m.

b. *Network Measurement Report* (NMR)

   Berdasarkan pada besar kecilnya sinyal (*Receiver Signal Strength*) yang diterima *handphone* yang ada di suatu "*sector cell*", maka posisi itu dapat ditentukan dengan lebih akurat.

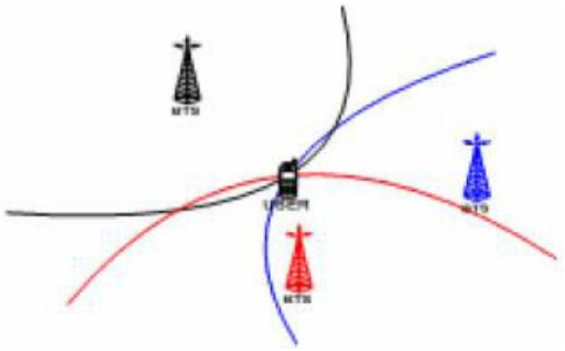


Gambar 2.8 Network Measurement Report [6]

2. Metode Enhanced Posittioning

   Metode ini menggunakan pendekatan *Observe Time Difference* (OTD) dengan jaringan GSM yang sering digunakan seperti *Enhanced*-OTD (E-OTD). E-OTD adalah metode pencarian posisi yang berdasarkan pada

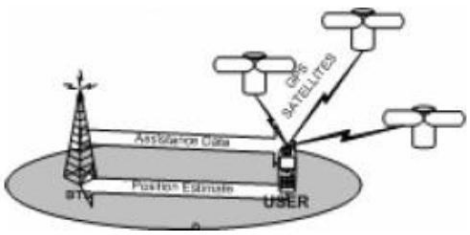
waktu. Untuk menentukan posisi *relative*, sebuah handphone harus aktif terhadap tiga *base station* dan perlu ditentukan terlebih dahulu jarak *handphone* terhadap masing – masaing *base station* berdasarkan waktu yang ditempuh oleh sebuah sinyal dari handphone ke masing – masing *base station*. Untuk menentukan posisi dari handphone yang sedang aktif digunakan rumus matematika untuk triangulasi. Dengan metode ini akurasinya akan meningkat hingga ketilitian kurang dari 50 m.



Gambar 2.9 Enhanced Positioning [5]

3. Metode *Advance Positioning*

   Metode ini menggunakan *teknologi Assisted-Global Positioning System* (A-GPS). A-GPS ini merupakan metode yang berbasis pada waktu. Pada metode ini, akan dilakukan pengukuran waktu tiba dari sebuah sinyal yang dikirim dari tiga buah satelit GPS. Hal ini berarti handset harus memiliki fasilitas untuk mengakses GPS. A-GPS juga menghasilkan akurasi secara *vertical* dan estimasi jarak yang baik. Akurasinya pun kurang dari 10 m.



Gambar 2.10 Advance Positioning [5]

2.6.2 Komponen *Location Based Service*

Dalam menggunakan *Location Based Service* atau layanan berbasis lokasi terdapat beberapa elemen yang diperlukan, antara lain :

1. *Mobile Devices*

   Sebuah alat yang digunakan untuk meminta informasi yang dibutuhkan seperti perangkat *mobile smartphone* yang memiliki fasilitas navigasi.
2. *Communication Network*

   Jaringan selular yang mengirimkan data pengguna dan permintaan layanan.
3. *Positioning Component*

   Untuk dapat mengolah layanan harus menentukan lokasi pengguna. Posisi pengguna ini dapat diperoleh menggunakan jaringan komunikasi atau dengan menggunakan *Global Positioning System* (GPS).
4. *Service and Application Provider*

   Penyedia layanan pengguna selular yang bertanggung jawab untuk memproses suatu layanan.
5. *Data and Content Provider*

   Penyedia layanan informasi data yang dapat diminta oleh pengguna.

2.6.3 *Latitude* (Garis Lintang)

Latitude adalah garis yang melintang di antara kutub utara dan kutub selatan yang menghubungkan sisi bagian bumi dari timur dan barat. Garis lintang memiliki posisi membentangi bumi seperti halnya garis khatulistiwa tetapi dengan kondisi nilai tertentu. *Latitude* dibedakan menjadi dua wilayah yaitu lintang utara dan selatan dimana nilai koordinat di bagian utara selalu positif dan koordinat selatan selalu negatif.

2.6.4 *Longitude* (Garis Bujur)

Longitude adalah garis bujur yang menghubungkan antara sisi utara dan sisi selatan permukaan bumi. Garis bujur digunakan untuk mengukur sisi barat

dan timur koordinat suatu titik di belahan bumi. *Longitude* dibedakan menjadi dua wilayah yaitu bujur timur dan bujur barat dimana nilai koordinat yang berada di timur selalu negatif dan nilai koordinat dibagian barat selalu positif.

2.7 GPS

Global Positioning System (GPS) merupakan sistem navigasi yang menggunakan satelit dalam penggunaannya. GPS terdiri dari tiga bagian yaitu sistem kontrol, satelit dan pengguna. Sistem kontrol adalah bagian yang mengontrol pergerakan satelit – satelit yang ada dan saling berinteraksi satu sama lain kemudian pengguna adalah alat navigasi yang digunakan seperti perangkat mobile yang kini sudah memiliki fitur GPS didalamnya. GPS biasanya digunakan untuk menunjukan suatu lokasi yang berada di permukaan bumi dengan tingkat akurasi yang cukup baik yaitu kurang dari 10 meter selama tidak ada benda padat yang dapat menghambat sinyal untuk mendapatkan lokasi pengguna.

2.8 Google *Maps*

Google *Maps* adalah sebuah layanan gratis dari Google tentang berbagai macam teknologi pemetaan. Google *Maps* menampilkan peta dunia secara digital sehingga lebih memudahkan ketika ingin mengetahui suatu lokasi tertentu. Google *Maps* menyediakan beberapa keunggulan seperti peta jalan, menunjukan rute untuk suatu perjalanan yang diinginkan baik itu melalui jalur mobil sampai pejalan kaki. Google maps ini dapat di akses di http://maps.google.com/.

2.9 Google *Maps* API

Google *Maps* API adalah sebuah layanan yang memungkinkan Google *Maps* agar dapat diintegrasikan ke dalam aplikasi yang memang membutuhkan fitur dari Google *Maps* ini. Google *Maps* API adalah sebuah *library Javascript*. Dengan menggunakan Google *Maps* API ini dapat memudahkan pengembang dalam membuat aplikasi yang membutuhkan teknologi peta di dalamnya dan tidak perlu memikirkan cara membuatnya secara manual karena dengan menggunakan

Google *Maps* API ini sudah semua yang dibutuhkan terkait teknologi peta digital sudah cukup lengkap dan dapat dimanfaatkan untuk aplikasi yang membutuhkannya.

Saat ini Google *Maps* API telah mengalami perkembangan, hingga saat ini Google Maps API telah mencapai versi 3. Namun, Google *Maps* API versi 3 ini masih hanya bisa digunakan oleh pemrograman berbasis web dengan menggunakan *Javascript*. Untuk penggunaan Google *Maps* API pada perangkat *mobile* seperti misalnya android Google Maps API yang bisa digunakan saat ini telah mencapai versi 2. Perkembangan pada Google Maps Versi 2 ini adalah dengan memiliki fitur – fitur yang lengkap dari versi sebelumnya salah satunya adalah dapat menampilkan peta dengan sudut yang diingkan dan berupa 3D *view*. Selain itu pada Google *Maps* API terdapat beberapa pilihan model peta yang ditampilkan seperti *Roadmap*, *Satellite, Terrain* dan *Hybrid*.

2.10 Web Service

Perangkat peranti lunak integrasi aplikasi perusahaan sifatnya spesifik terhadap suatu produk, artinya perangkat ini hanya dapat bekerja dengan bagian-bagian peranti lunak dan sistem operasi tertentu. Contoh, suatu perangkat EAI untuk menghubungkan bagian tertentu dari peranti lunak pencatatan penjualan untuk aplikasi produksi, pengangkutan, dan penagihan, tidak dapat bekerja dengan peranti lunak pencatatan penjualan dari vendor lainnya. Program *middleware* yang dikembangkan oleh sistem BEA, sebuah integrator yang besar, tidak dapat berhubungan dengan aplikasi *middleware vendor* lainnya yang dibeli pada tahun-tahun sebelumnya tanpa pengeluaran yang besar untuk pemrograman dan perancangan. Layanan web (*web service*) mencoba memberikan alternatif standar untuk menghadapi masalah-masalah integrasi seperti itu dengan menciptakan sebuah lingkungan komunikasi yang netral.

Layanan web (*web service*) adalah serangkaian komponen peranti lunak yang bertukar informasi antara satu sama lain dengan bebas menggunakan komunikasi web dan bahasa yang standar. Layanan web dapat bertukar informasi antara dua sistem yang berbeda, terlepas dari sistem operasi atau bahasa

pemrograman sistem tersebut. Layanan web dapat dipadukan untuk membangun sistem penghubung aplikasi berbasis web standar terbuka dari dua organisasi yang berbeda, dan juga dapat digunakan untuk membuat aplikasi yang menghubungkan sistem terpisah dalam satu perusahaan. Layanan web tidak terikat dengan sistem operasi atau bahasa pemrograman apa pun, dan aplikasi-aplikasi yang berbeda dapat digunakan untuk berkomunikasi satu sama lain dengan cara standar tanpa penulisan kode khusus yang menghabiskan waktu.

2.11 JSON (Javascript Object Notation)

JSON atau *Javascript Object Notation* adalah format untuk pertukaran data yang lebih ringan,mudah dibaca dan ditulis oleh manusia serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. JSON lebih unggul dari pada format pertukaran data yang lain seperti XML. Mulai dari kecepatan, penulisan yang lebih mudah dan *coding parsing* yang lebih ringkas dan sederhana. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman lain / apapun karena menggunakan bahasa pemrograman yang umum digunakan yaitu keluarga C seperti C, C#, C++, Java, Javascript, Perl, Python dan lain – lain. [7]

2.12 Pemodelan Basis Data

2.12.1 Pengenalan Basis Data

Kata basis data atau *database* biasa digunakan untuk menguraikan segala sesuatu dari sekumpulan data tunggal, seperti daftar telepon. Istilah basis data tidak termasuk aplikasi, yang terdiri dari *form* dan *report* dimana pengguna akan saling berhubungan. Basis data terdiri dari file-file fisik yang ditetapkan berdasarkan komputer saat menerapkan perangkat lunak basis data. Di sisi lain, suatu model basis data lebih kepada konsep dibandingkan objek fisik dan digunakan untuk menciptakan tabel dalam basis data.

Basis data adalah suatu aplikasi terpisah yang menyimpan suatu koleksi data. Masing-masing basis data memiliki satu API atau lebih yang berbeda untuk menciptakan, mengakses, mengelola, mencari, dan mereplikasi data.

Sebuah basis data adalah tempat penyimpanan file data. Sebagai file data, suatu basis data tidak menyajikan informasi secara langsung kepada pengguna. Pengguna harus menjalankan aplikasi untuk mengakses data dari basis data dan menyajikannya dalam bentuk yang bisa dimengerti. Ketika bekerja dengan file-file data, suatu aplikasi harus dikodekan agar bekerja dengan struktur masing-masing file data. Biasanya, suatu basis data berisi suatu katalog yang menggunakan aplikasi untuk menentukan cara data diorganisir. Aplikasi basis data umum bisa menggunakan katalog tersebut untuk menampilkan data dengan pengguna dari basis data yang berbeda secara dinamis, tanpa terikat pada format data tertentu.

2.12.2 Database Management System (DBMS)

Database Management System (DBMS) atau sistem manajemen basis data adalah suatu sistem perangkat lunak kompleks yang mengatur permintaan dan penyimpanan data ke dan dari *harddisk*. DBMS menyediakan keamanan (*security*), privasi (*privacy*), integritas (*integrity*), dan kontrol konkurensi (*concurrency controls*). DBMS mengelola transaksi pada *multiuser*, lingkungan akses bersamaan, dan menyediakan tingkat independensi data yang mengisolasi pandangan (*view*) pengguna atau aplikasi dari perubahan yang berlangsung di tingkat internal dan konseptual.

DBMS adalah perangkat lunak yang disajikan oleh penjual basis data. Produk perangkat lunak seperti Microsoft Access, Oracle, Microsoft SQL Server, Sybase, DB2, INGRES, dan MySQL adalah DBMS.

DBMS menyediakan semua layanan dasar yang diperlukan untuk mengorganisir dan memelihara basis data, termasuk layanan berikut:

a. Memindahkan data ke dan dari file-file data fisik jika dibutuhkan.
b. Mengelola akses data oleh berbagai pengguna secara bersamaan; mencakup ketentuan untuk mencegah peng-*update*-an secara bersamaan.
c. Mengelola transaksi sehingga perubahan transaksi basis data adalah semua atau tidak sama sekali. Dengan kata lain, jika transaksi berhasil, semua basis data yang dibuat berubah dan direkam di dalam basis data. Jika

transaksi gagal, tidak satupun dari perubahan tersebut yang direkam oleh basis data.

d. Mendukung bahasa query (*query language*), yang mana suatu sistem perintah mempekerjakan pengguna basis data untuk mendapatkan data kembali dari basis data.
e. Ketentuan untuk mem-*backup* basis data dan pemulihan dari kegagalan.
f. Mekanisme keamanan untuk mencegah perubahan dan akses data yang tidak sah.

2.12.3 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau yang dikenal dengan DBMS (*Database Management System*), *database* ini *multithread*, *multiuser*.

Kekuatan MySQL tidak ditopang oleh sebuah komunitas, seperti Apache, yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh pemilik masing-masing, tetapi MySQL didukung penuh oleh sebuah perusahaan profesional dan komersial, yakni MySQL AB dari Swedia.

MySQL adalah *Relational Database Management System* (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). Dimana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat *closed source* atau komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Sebagai *database server*, MySQL dapat dikatakan lebih unggul dibandingkan *database server* lainnya, terutama dalam kecepatan. Berikut ini beberapa keistimewaan MySQL, antara lain:

a. Probability

MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.

b. *Multiuser*

MySQL dapat digunakan oleh beberapa *user* dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

c. *Security*

MySQL memiliki beberapa lapisan sekuritas seperti level *subnetmask*, nama *host*, dan izin akses *user* dengan sistem perizinan yang mendetail serta *password* terenkripsi.

d. *Scalability* dan *limits*

MySQL mampu menangani *database* dalam skala besar, dengan jumlah *records* lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.

2.12.4 Model Entity-Relationship

Model *Entity-Relationship* (ER) mula-mula diusulkan oleh Peter Chen pada tahun 1976 sebagai cara untuk mempersatukan pandangan basis data jaringan dan relasional. Langkah sederhana dari model ER adalah model data dan konseptual yang memandang dunia nyata sebagai kesatuan (entities) dan hubungan (relationships). Komponen dasar model merupakan diagram entity-relationship yang digunakan untuk menyajikan objek data secara visual. Chen menulis bahwa model telah diperluas dan kini bisa digunakan untuk merancang basis data. Keunggulan model ER dalam perancangan tersebut adalah:

1. Mampu memetakan model relasional dengan baik. Pembangunan yang digunakan di dalam model ER dengan mudah diubah ke dalam tabel relasional.
2. Sederhana dan mudah dipahami hanya dengan sedikit pelatihan. Oleh karena itu, model bisa digunakan oleh perancang basis data untuk mengkomunikasikan perancangan kepada pengguna akhir.

3. Sebagai tambahan, model bisa digunakan sebagai suatu rencana perancangan oleh pengembang basis data untuk menerapkan suatu model data dalam perangkat lunak manajemen basis data spesifik.

Perancangan basis data dengan menggunakan model *entity-relationship* adalah dengan menggunakan *Entity Relationship Diagram* (ERD). ERD mengilustrasikan struktur logis dari basis data.

2.12.5 Model Relasional

Model relasional secara formal diperkenalkan oleh Dr. E. F. Codd pada tahun 1970 dan berkembang sejak saat itu melalui serangkaian tulisan. Model relasional menyajikan data berbentuk tabel dua dimensi (*two-dimension*). Masing-masing tabel menyajikan beberapa orang, tempat, sesuatu hal, atau peristiwa sebagai informasi yang dikumpulkan. Suatu basis data relasional adalah suatu kumpulan tabel *two-dimensional*.

Model relasional merupakan model data yang paling banyak digunakan saat ini. Tingkatan-tingkatan data yang menyusun database disebut skema, dimana setiap tingkatan menggambarkan sebuah data. Model relasional ini memfokuskan pada pembahasan hubungan (relasional) antardata. Skema relasi pada model relasional ditentukan oleh nama skema, nama, dan tipe field atau kolom. Model data relasional merepresentasikan datanya menggunakan bentuk tabel-tabel yang saling berelasi/berhubungan dengan suatu *field* kunci. Setiap tabel memiliki banyak kolom dengan nama yang unik untuk setiap kolomnya. Model relasi menempati level abstraksi yang lebih rendah daripada model ER, karena sifatnya langsung menunjukkan bentuk *record* yang akan disimpan dalam suatu file.

2.13 Pemrograman Web

2.13.1 HTML

HTML adalah singkatan dari *Hyper Text Markup Language*. HTML digunakan untuk membuat halaman web. Sebuah file dokumen yang ditulis dalam

format HTML akan dibaca dan diterjemahkan oleh web *browser* (misal Internet Explorer) untuk kemudian disajikan dalam bentuk web.

File-file yang ditulis dalam format HTML disimpan dengan ekstensi *.htm* atau *.html*. File-file tersebut dapat ditulis/diketik menggunakan berbagai macam teks editor, misalkan Notepad, Wordpad, dan lain sebagainya.

HTML terdiri atas berbagai macam Tag yang digunakan untuk menandai dan mengatur tampilan halaman *web* yang dibuat. Misalkan jika ingin membuat sebuah kalimat bercetak tebal di *web*, maka gunakan Tag yang cocok untuk itu.

Tag biasanya dituliskan berpasangan di awal dan di akhir bagian file yang akan ditandai. Untuk Tag yang diletakkan di awal bagian ditulis dengan tanda lebih kecil (<), nama Tag, dan tanda lebih besar (>). Dan untuk Tag yang diletakkan di akhir bagian ditulis dengan tanda lebih kecil (<), garis miring (/), nama Tag, dan tanda lebih besar (>).

2.13.2 CSS

CSS (*Cascading Style Sheet*) adalah suatu teknologi yang digunakan untuk memperindah tampilan halaman *website* (situs). Singkatnya dengan menggunakan metode CSS ini dapat dengan mudah mengubah secara keseluruhan sekaligus memformat ulang situs. CSS dapat digunakan untuk menghindari pekerjaan berulang-ulang.

CSS mempunyai dua bagian utama yaitu *selector* dan deklarasi. Yang disebut *selector* biasanya elemen html yang ingin di ubah, sedangkan deklarasi biasanya terdiri dari *property* dan nilai, *property* sendiri adalah atribut *style* yang ingin di ubah, dan setiap *property* memiliki nilai. Setiap *syntax* css selalu diakhiri dengan titik koma (;), dan kelompok deklarasi dikelilingi oleh kurung kurawal ({}).

Ada dua cara untuk membuat aturan *style*, yaitu dengan menuliskannya langsung dalam dokumen HTML atau dengan memisahkannya dalam sebuah file khusus yang berekstensi *.css*. Cara yang pertama dapat dilakukan dengan menambahkan Tag <STYLE> diantara Tag <HEAD> dan </HEAD>. Kemudian tuliskan aturan stylenya di dalam Tag <STYLE> tersebut. Cara pertama sudah

jarang dilakukan. Saat ini, para pembuat web biasanya menggunakan cara yang kedua, yaitu membuat aturan *style* dalam file terpisah.

2.13.3 PHP

PHP adalah singkatan dari PHP: *Hypertext Preprocessor*. PHP merupakan bahasa pemrograman untuk membuat web yang bersifat *server-side scripting*. PHP memungkinkan untuk membuat halaman web yang bersifat dinamis.

PHP dapat dijalankan pada berbagai macam sistem operasi, misalnya Windows, Linux, dan Mac OS. Selain Apache, PHP juga mendukung beberapa *web server* lain, misalnya Microsoft IIS, Caudium, PWS, dan lain-lain.

PHP dapat memanfaatkan *database* untuk menghasilkan halaman *web* yang dinamis. Sistem manajemen *database* yang sering digunakan bersama PHP adalah MySQL. Namun, PHP juga mendukung sistem manajemen *database* Oracle, Microsoft Access, Interbase, dBase, PostgreSQL, dan lain-lain.

Hingga kini, PHP sudah berkembang hingga versi 5. PHP 5 mendukung penuh *Object-Oriented Programming* (OOP), integrasi XML, mendukung semua ekstensi terbaru MySQL, pengembangan *web services* dengan SOAP dan REST, serta ratusan peningkatan lainnya dibandingkan versi sebelumnya. PHP juga bersifat *open source* sehingga setiap orang dapat menggunakannya secara gratis.

2.13.4 JavaScript

JacaScript merupakan bahasa *scripting* (bahasa pemrograman yang ringan) yang popular di internet dan dapat bekerja si sebagian besar *browser* popular seperti Internet Explorer (IE), Mozilla FireFox, Netscape, Google Chrome dan Opera. Kode JavaScript dapat disisipkan dalam halaman web menggunakan tag <SCRIPT>.

JavaScript adalah bahasa interpreter yang berarti skrip dieksekusi tanpa proses kompilasi. JavaScript berisi baris kode yang dijalankan di komputer (*web browser*). JavaScript didesain untuk menambah interaktif suatu *web*.

Ada beberapa cara untuk menuliskan JavaScript, yaitu di bagian <HEAD>, <BODY> atau external. JavaScript dalam sebuah halaman *web* akan

dieksekusi saat halaman di *load* oleh *browser*. Namun, hal ini kadang tidak sesuai dengan keinginan karena suatu saat akan mengeksekusi skrip saat halaman di *load*, tetapi kadang JavaScript dijalankan melalui *trigger* sebuah *event*.

2.14 Pemrograman Java

2.14.1 Pengenalan Bahasa Pemrograman Java

Java merupakan bahasa pemrograman yang menanjak popularitasnya dewasa ini. Selain gratis, Java mudah didapatkan dan juga tangguh. Java sendiri lahir pada tahun 1991 yang diprakarsai oleh tim Sun Microsystems melalui proyek bernama Green yang dipimpin oleh James Gosling.

Java menerapkan fitur-fitur dari bahasa pemrograman lain, dimana fitur itu dianggap merupakan siatu kelebihan oleh tim Sun. Misalnya JVM/JRE yang merupakan mesin maya pada bahasa Pascal, Sintaks, dan *Exception Handling* diambil dari C/C++ dan lain sebagainya.

Java Virtual Machine (JVM) atau *Java Runtime Environment* (JRE) merupakan fitur Java yang membuatnya dapat berjalan pada semua *platform* sistem operasi. Apabila membuat Java pada Windows, dapat pula dijalankan pada sistem operasi Linux, Mac OS, dan lainnya.

Java memiliki beberapa keunggulan yang tidak dimiliki oleh bahasa lain. Berdasarkan *white paper* resmi dari Sun Microsystems, Java memiliki karakteristik sebagai berikut:

1. Sederhana

   Bahasa pemrograman Java menggunakan sintaks mirip C++, namun pada Java telah banyak diperbaiki terutama menghilangkan penggunaan *pointer* yang rumit dan *multiple inheritance*. Java juga menggunakan *automatic memory allocation* dan *memory garbage collection*.
2. Berorientasi objek (*object oriented*)

   Java menggunakan pemrograman berorientasi objek yang membuat program dapat dibuat secara modular dan dapat dipergunakan kembali.

3. Dapat didistribusi dengan mudah

   Java dibuat untuk membuat aplikasi terdistribusi secara mudah dengan adanya *libraries networking* yang terintegrasi pada Java.

4. *Interpreter*

   Program Java dijalankan menggunakan *intrepeter*, yaitu *Java Virtual Machine* (JVM). Hal ini menyebabkan *source code* Java yang telah dikompilasi menjadi Java *bytecodes* dapat dijalankan pada *platform* yang berbeda-beda.

5. Robust

   Java mempunyai reliabilitas yang tinggi. *Compiler* pada Java mempunyai kemampuan mendeteksi *error* secara lebih teliti dibandingkan bahasa pemrograman lain. Java mempunyai *runtime-Exception handling* untuk membantu mengatasi *error* pada pemrograman.

6. Aman

   Sebagai bahasa pemrograman untuk aplikasi internet dan terdistribusi, Java memiliki beberapa mekanisme keamanan untuk menjaga aplikasi tidak digunakan untuk merusak sistem komputer yang menjalankan aplikasi tersebut.

7. *Architecture Neutral*

   Pemrograman Java merupakan *platform independent*. Program cukup mempunyai satu buah versi yang dapat dijalankan pada *platform* yang berbeda dengan *Java Virtual Machine*.

8. Portabel

   Source code maupun program Java dapat dengan mudah dibawa ke *platform* yang berbeda-beda tanpa harus dikompilasi ulang.

9. *Performance*

   Performance pada Java sering dikatakan kurang tinggi. Namun *performance* Java dapat ditingkatkan menggunakan kompilasi Java lain seperti Inprise, Microsoft, ataupun Symantec yang menggunakan *Just In Time Compilers* (JIT).

10. *Multithread*

    Java mempunyai kemampuan untuk membuat suatu program yang dapat melakukan beberapa pekerjaan secara sekaligus dan simultan.

11. Dinamis

    Java didesain untuk dapat dijalankan pada lingkungan yang dinamis. Perubahan pada suatu *class* dengan menambahkan *properties* ataupun *method* dapat dilakukan tanpa menggangu program yang menggunakan *class* tersebut.

2.14.2 Eclipse

Eclipse merupakan sebuah IDE (*Integrated Development Environment*) untuk mengembangkan suatu aplikasi dan dapat dijalankan di berbagai platform (Platform-Independent). Eclipse termasuk kedalam tools yang bersifat *open source*. Eclipse memiliki beberapa sifat diantaranya adalah sebagai berikut :

1. *Multi-platform*

   Target *system* operasi eclipse ini adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.

2. *Multi-language*

   Dikembangkan dengan bahasa pemrograman Java tetapi eclipse juga mendukung pengembangan aplikasi dengan bahasa pemrograman lain seperti C/C++, Cobol, Python, Perl, PHP dan lain – lain.

3. *Multi-role*

   Eclipse dapat digunakan untuk aktifitas dalam siklus pengembangan aplikasi sepertidokumentasi, pengujian, pengembangan dan sebagainya

2.14.3 Arsitektur Eclipse

Eclipse merupakan sebuah kernel yang berarti bahwa eclipse adalah fungsi dari pluf-in yang sudah dipasang (di-*install*). Basis dari eclipse dinamakan *Rich Client Platform* (RCP). Berikut adalah komponen yang membentuk RCP :

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

Eclipse dilengkapi dengan JDT (*Java Development Tools*), plug-in yang membuat eclipse dapat kompatibel untuk pengembangan program Java dan PDE (*Plug-in Development Environment*). Eclipse beserta plug-in yang dimilikinya diimplementasikan dalam bahasa pemrograman Java.

Eclipse tidak hanya untuk mengembangkan program Java, tetapi juga untuk berbagai macam keperluan yang dibutuhkan. Untuk melakukan perluasan apapun didalam eclipse cukup dengan meng-install plug-in yang dibutuhkan.

2.14.4 Versi Eclipse

Eclipse Foundation telah mengkoordinasikan peluncuran Eclipse secara rutin yang dikenal dengan nama simultaneous Release. Setiap versi yang diluncurkan terdiri dari eclipse platform dan sejumlah proyek yang terlibat langsung dalam pembuatan proyek eclipse.

Hampir setiap tahun ke tahun eclipse selalu dijadwalkan untuk meluncurkan versi terbarunya. Tujuannya adalah untuk menyediakan distribusi eclipse dengan fitur – fitur dan versi yang sudah terstandarisasi. Berikut berbagai versi eclipse yang sudah diluncurkan saat ini, yaitu :

Tabel 2.1 Versi Eclipse [7]

Kode Peluncuran	Tanggal Peluncuran	Platform	Nama Proyek
Eclipse 3.0	21 Juni 2004	3.0	-
Eclipse 3.1	28 Juni 2005	3.1	-
Callisto	30 Juni 2006	3.2	Callisto Project
Europa	29 Juni 2007	3.3	Europa Project
Ganymede	25 Juni 2008	3.4	Ganymede Project
Galileo	24 Juni 2009	3.5	Galileo Projecy
Helios	23 Juni 2010	3.6	Indigo Project
Juno	27 Juni 2012	4.2	Juno Project

2.14.5 Development Kit (JDK)

Java Development Kit (JDK) merupakan produk dari Sun Microsystems yang ditujukan untuk pengembang Java. Sejak Java diperkenalkan, hingga saat ini banyak yang telah menggunkan Java SDK. Pada tanggal 17 November 2006 Sun Microsystems mengumumkan bahwa akan dirilis JDK dibawah lisensi GNU (General Public License) sehingga membuat penggunaannya menjadi gratis.