BAB II

LANDASAN TEORI

2.1 Distro (*Distributionstore*)

Banyak pendapat soal pengertian distro. Ada yang bilang disto itu kependekan dari distribution outlet, ada yang bilang juga diambil dari distribution shop atau store. Dulunya distro hanya toko yang menjual merchandise band-band indie/underground. Merchandise tersebut dijual buat mendukung hidupnya band-band tersebut. Konsep awalnya adalah DIY (*Do It Yourself*), bikin sendiri, jual sendiri di toko sendiri.

Sekarang Distro sudah mengalami pergeseran makna karena barang yang dijualnya tidak lagi hanya barang merchandise band. Seperti halnya Distro-distro yang ada di Bandung, barang-barang yang dijual beraneka ragam seperti tas, kaos, jaket dan lainnya selalu *up to date*.

Dengan hal seperti diatas banyak kemungkinan dari luar daerah adanya ketertarikan untuk berbelanja ke distro-distro tersebut, tetapi keterjangkaun informasi dan mahalnya sebuah informasi hal tersebut kemungkinan tidak terpikirkan semua distro-distro.

2.2 Internet

Internet sebagai suatu bentuk jaringan komputer mempunyai beberapa keunikan yang tidak dipunyai oleh suatu bentuk jaringan komputer yang lain yang ada pada saat ini. Internet terdiri dari suatu kelompok jaringan (yang terus

tumbuh) yang teridiri dari baik jaringan publik maupun jaringan private. *Local Area Networks* (*LANs*), *wide area Networks* (*WANs*) yang sama lain saling terhubung (*interconnect*). Secara teknis internet adalah " *a network of many networks, all running the TCP/IP protocol suite.., connected through gateways and sharing common name and address spaces*". Adanya interkoneksi dengan berbagai jaringan inilah yang menyebabkan internet berbeda dengan jaringan yang ada lainnya. Interkoneksi inilah yang menghubungkan setiap orang dimanapun ia berada di dunia ini dapat mengakses internet baik melalui jaringan local, regional ataupun internasional sehingga ia dapat berkomunikasi dengan orang lain dimanapun orang itu berada.

Jaringan-jaringan (*networks*) yang saling terhubung itu, berkomunikasi satu sama lainnya dengan suatu teknologi komunikasi yang terdiri suatu kelompok protocol yang secara singkat disebut dengan TCP/IP (*Transper Control Protocol/Internet Protocol*)

TCP/IP pertamakali dikembangkan oleh US Departement of Defense, maksud dari pengembangannya adalah untuk membuat suatu jaringan komputer militer yang dapat tetap terhubung satu sama lainnya meskipun terdapat salah satu komponennya rusak dalam sauatu serangan militer. TCP/IP berfungsi untuk membagi-bagi data digital menjadi paket-paket kecil (*datagram*) yang kemudian akan ditransmisikan melalui melalui jaringan ke suatu tujuan yang dikehendaki. Rute maupun jenis jaringan/perangkat jaringan yang akan dilewati oleh data tersebut dapat melalui rute yang berada satu sama lainnya. Jika kita mengirim suatu surat/pesan melalui internet, TCP akan akan membagi pesan kedalam paket-

paket data yang kecil. Setiap paket data tersebut akan ditandai dengan nomer urut dan alamat penerima. Selain TCP akan mengertakan inforamasi untuk mengontrol jika terjadi kesalahan.

Paket-paket data tersebut dikirim melalui jaringan komputer, dimana dalam tahap ini IP membawa paket-paket data tersebut dan memerikasanya jika tejadi kesalahan. Setelah semua paket data itu dapat diterima secara benar, TCP menggunakan nomer urut tersebut untuk merekonstruksi pesan asli. Berdasarkan uraian tersebut maka fungsi TCP adalah mengatur paket-paket data dan memastikan kebenaran paket data. Sedangkan fungsi dari IP adalah membawa paket-paket data dari suatu tempat ke tempat yang lainnya.

Internet menghubungkan (*interconnect*) berbagai jaringan yang ada di dunia. Kemampuan ini menjadikan internet sebagai suatu jaringan public yang mempunyai sifat terbuka (*Open & Public Networks*). Jaringan terbuka (Open network) ini mempunyai berbagai keuntungan dibandingkan jaringan yang tertutup (*close network*).

2.3 Intranet dan Ekstranet

Bagi organisasi/perusahaan yang membutuhkan komunikasi yang bersifat lebih pribadi/rahasia dapat membangun jaringan dengan teknologi yang sama tetapi terpisah dari Internet. Saat ini telah dikembangkan konsep komunikasi “baru” dengan nama Intranet dan Ekstranet. Secara singkat, teknologi Intranet memanfaatkan program-program Internet untuk keperluan eksplorasi/publikasi data dan komunikasi yang terbatas pada kompleks perkantoran suatu organisasi

sedangkan Ekstranet menggabung jaringan-jaringan Intranet kantor pusat dan cabang-cabang dengan memanfaatkan saluran data umum (Internet) tanpa memberi servis apapun pada pengguna Internet lainnya.

2.4 Sistem Informasi

Istilah TI (Teknologi Informasi) atau IT (*Information Technology*) yang populer saat ini adalah bagian dari mata rantai panjang dari perkembangan istilah dalam dunia SI (Sistem Informasi) atau IS (*Information System*). Istilah TI memang lebih merujuk pada *teknologi* yang digunakan dalam menyampaikan maupun mengolah informasi, namun pada dasarnya masih merupakan bagian dari sebuah sistem informasi itu sendiri. TI memang secara *nota bene* lebih mudah dipahami secara umum sebagai pengolahan informasi yang berbasis pada *teknologi* komputer yang tengah terus berkembang pesat. Sebuah Sistem TI atau selanjutnya akan disebut STI, pada dasarnya dibangun di atas lima tingkatan dalam sebuah piramida STI. Berurutan dari dasar adalah : konsep dasar, teknologi, aplikasi, pengembangan dan pengelolaan.

2.5 Pengembangan Sistem Informasi

Secara umum untuk menciptakan suatu perangkat lunak atau sistem informasi, tidak luput adanya suatu motede pengembangan seperti gambar 1.1 pada bab 1, yang dilakukan dengan pengembangan berbasis objek dan disamping itu juga setelah selesai melakukan pengembangan adanya suatu pendekatan dalam

perancangan sistem. Perancanga tersebut biasanya dilakuka dengan metode pendekatan terstruktur yang '*tool*' utamanya adalah DFD (*Data Flow Diagram*).

Pendekatan terstruktur memiliki '*tool-tool*' perancangan yang dikenal secara luas serta menjadi standar umum, seperti DFD (*Data Flow Diagram*), Bagan tersruktur (*Structured Chart*), Diagram Alir (*Flow Chart*) dan sebagainya. Sayang sekali, karena metodologi berorientasi objek adalah yang relatif baru, ia belum memiliki standar '*tool*' untuk analisis dan perancangan. Walau pun begitu, penulis mencoba menarapakkan pada metode UML (*Unified Modeling Language*) yang telah berangsur-angsur menjadi standar metodologi pengembangan perangkat lunak/ atau sistem informasi berorientasi objek.

2.5.1 UML (*Unified Modeling Language*)

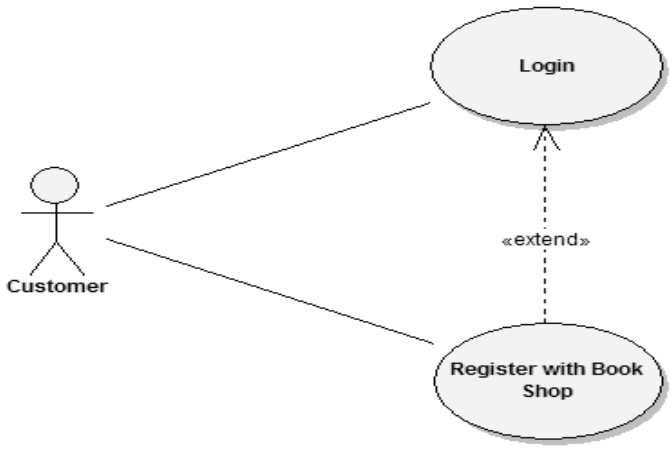
Secara umum UML merupakan 'bahasa' untuk visualisasi, spesifikasi, konstruksi, serta dokumentasi. Dalam kerangka visualisasi, para pengembang menggunakan UML sebagai suatu cara untuk mengkomunikasikan idenya kepada para pemrogram serta calon pengguna system/perangkat lunak. Dengan adanya 'bahasa' yang bersifat standar, komunikasi perancangan dengan pemrogram (lebih tepat lagi: komunikasi antaranggota kelompok pengembang) serta calon pengguna diharapkan menjadi mulus. Dalam kerangka spesifikasi, UML menyediakan model-model yang tepat, tidak mendua-arti (*ambigiu*), serta lengkap. Secara khusus, UML menspesifikasikan langkah-langkah penting dalam pengambilan keputusan analisis, perancangan, serta implementasi dalam sistem yang sangat bernuansa perangkat lunak (*software intensive system*).

Pendekatan yang dilakukan dengan metode UML (*Unified Modeling Language*) penulis membuat beberapa abstrak konsep dasar dari UML, yaitu terdiri dari *structural classification*, *dynamic behavior*, dan *model management*. Untuk penentuan pertama dalam pembuatan UML penulis mendefenisikan dalam diagram-diagram yang nantinya diperlukan, yaitu :

1. *use case diagram*

use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah system. Yang ditekankan pada “apa” yang diperbuat system, dan bukan “bagaimana”. Sebuah *use case diagram* mempresentasikan sebuah interaksi antara actor dengan system. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke system, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah actor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan system untuk melakukan pekerjaan-pekerjaan tertentu.

Contoh penggunaan *use case diagram* :



Gambar 2.1 Contoh penggunaan *use case diagram*

2. *class diagram*

Class merupakan sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggambarkan keadaan (atribut/properti) suatu system, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metode/fungsi).

Class diagram meggambarkan struktur dari deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan asosiasi, dan lain-lain.

Class memiliki tiga area pokok :

1. Nama (dan stereotype)
2. Atribut
3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

a. *Private,* tidak dapat dipanggil dari luar *class* yang bersangkutan
b. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya
c. *Public,* dapat dipanggil oleh siapa saja.

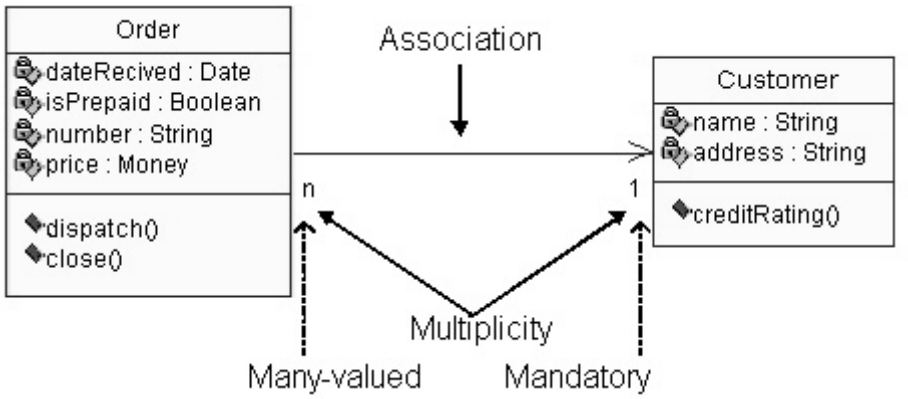
Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memilikimetoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class.* Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*.

Sesuai dengan perkembangan *class* model, *class* dapat dikelompokkan menjadi *package*. Kita juga dapat membuat diagram yang terdiri atas *package*.

Hubungan Antar *Class*

1. Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. Panah *navigability* menunjukkan arah *query* antar *class*.
2. Agregasi, yaitu hubungan yang menyatakan bagian (“terdiri atas..”).
3. Pewarisan, yaitu hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.
4. Hubungan dinamis, yaitu rangkaian pesan (*message*) yang di-*passing* dari satu *class* kepada*class* lain. Hubungan dinamis dapat digambarkan dengan menggunakan *sequence diagram*yang akan dijelaskan kemudian.

Contoh penggunaan *class diagram* :



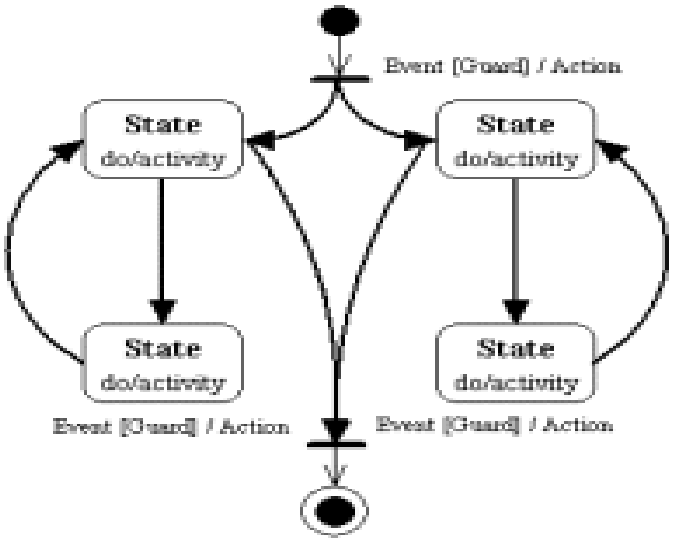
Gambar 2.2 Contoh penggunaan class diagram

3. *statechart diagram*

statechart diagram menggambarakan transisi dan perubahan keadaan (dari satu *state* ke *state* lainnya) suatu objek pada sistem sebagai akibat dari *stimuli* yang diterima. Pada umumnya *statechart diagram* menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu *statechart* diagram).

Dalam UML, *state* digambarkan berbentuk segiempat dengan sudut membulat dan memiliki nama sesuai kondisinya saat itu. Transisi antar state umumnya memiliki kondisi guard yang merupakan syarat terjadinya transisi yang bersangkutan, dituliskan dalam kurung siku. *Action* yang dilakukan sebagai akibat dari event tertentu dituliskan dengan diawali garis miring.

Contoh penggunaan *statechart diagram* :



Gambar 2.3 Contoh penggunaan *statechart diagram*

4. *activity diagram*

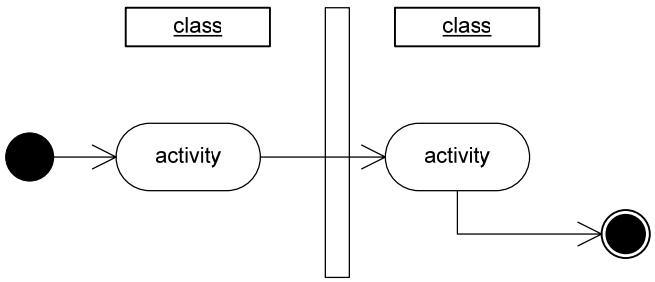
menggambarkan berbagai alir aktivitas dalam system yang sedang dirancang. Bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka terakhir. *Activity diagram* juga dapat menggambarkan proses *parallel* yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan *state diagram* khusus, dimana sebagian besar *state* adalah *action* dan sebagai besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan behavior internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

Contoh penggunaan *Activity diagram* :



Gambar 2.4 Contoh penggunaan *Activity diagram*

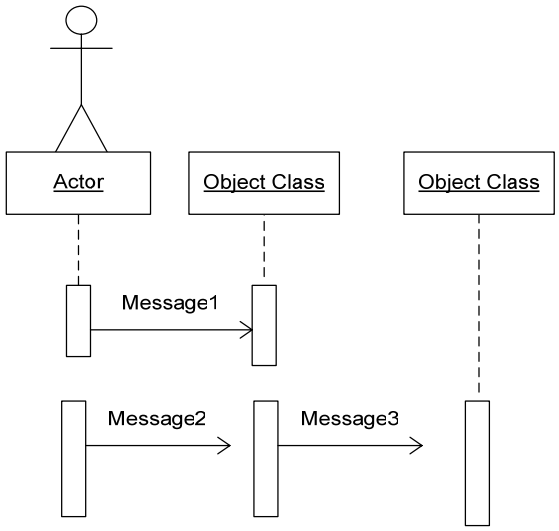
5. *sequence diagram*

Sequence diagram mengambarkan interaksi antar objek didalam dan disekitar sistem (termasuk pengguna, display, dan sebagainya) berupa *massage*

yang digambarkan terhadap waktu. *sequence diagram* terdiri dari antar dimensi vertical (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan scenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan.

Contoh Penggunaan *Sequence diagram :*

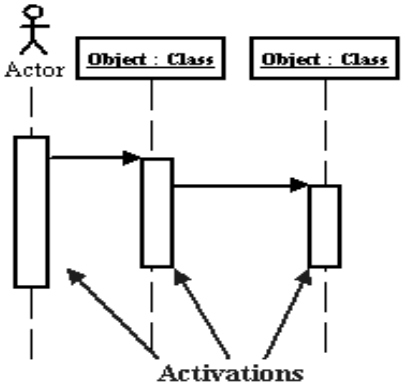


Gambar 2.5 Contoh penggunaan *Sequence diagram*

6. *collaboration diagram*

Collaboration diagram juga menggambarkan interaksi antar objek seperti *sequence diagram*, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message*. Setiap *massage* memiliki *sequence number*, dimana *message* dari level tertinggi memiliki nomor 1. Messages dari level yang sama memiliki prefiks yang sama.

Contoh Penggunaa *Collaboration diagram* :



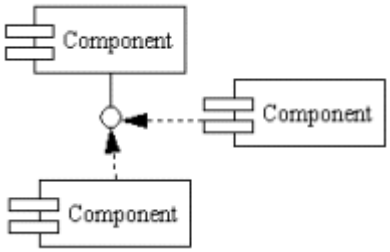
Gambar 2.6 Contoh Penggunaan *Collaboration diagram*

7. *component diagram*

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) diantaranya komponen piranti lunak adalah modul berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time*, *link time*, maupun *run time*. Umumnya komponen terbentuk dari beberapa class dan/atau *package*, tetapi dapat juga dari komponen-komponen yang lebih kecil.

Komponen dapat juga berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.

Contoh penggunaan *Component diagram* :

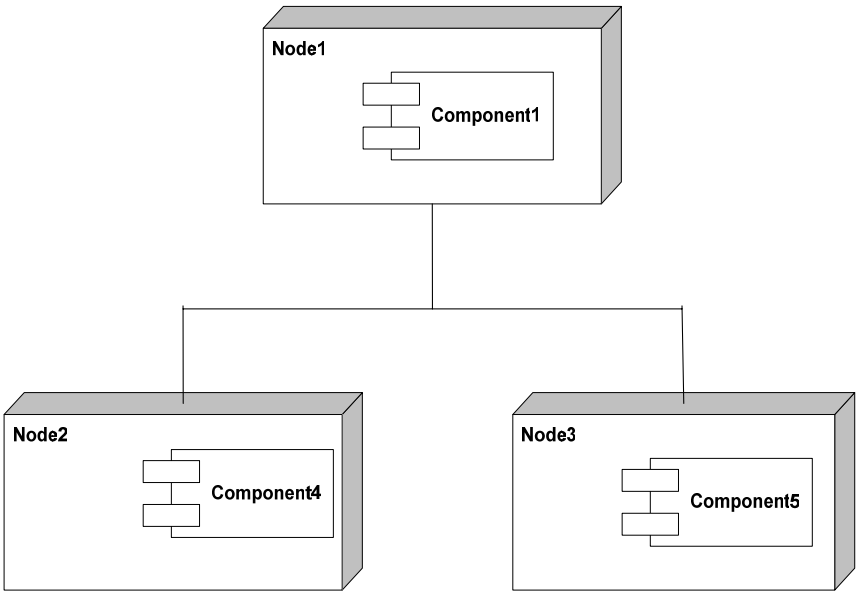


Gambar 2.7 Contoh Penggunaan *Component diagram*

8. *deployment diagram*

Deployment diagram menggambarkan detail bagaiman komponen di-*deploy* dalam infrastruktur sistem, diman komponen akan terletak (pada mesin, server atau piranti keras apa), bagaiman kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal yang lain bersifal fisikal.

Contoh Penggunaan *Deployment diagram* :



Gambar 2.8 Contoh Penggunaan *Deployment diagram*

2.6 Basisdata (*Database*)

Basisdata adalah kumpulan data (elementer) yang secara logic berkaitan dalam mempresentasikan fenomena/fakta secara terstruktur dalam domain tertentu untuk mendukung aplikasi pada system tertentu.

Database dapat didefenisikan dalam sejumlah sudut pandang seperti :

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan sedemikian rupa tanpa perulangan (*redudansi*) yang tidak perlu.
3. Kumpulan *file* table/arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis.

2.6.1 DBMS (*Database Manajemen System*)

Sistem manajemen basisdata atau DBMS (*Database Manajemen System*) adalah perangkat lunak untuk mendefenisikan, menciptakan, mengelola dan mengendalikan pengaksesan basisdata. Fungsi manajemen system basisdata saat ini yang paling penting adalah menyediakan basis untuk sistem informasi manajemen.

Tujuan utama sistem manajemen basisdata adalah menyediakan lingkungan yang nyaman dan efesien untuk penyimpanan dan pengambilan data dari basisdata. Sistem manajemen basisdata berperan memberi abstrak dan tingkat tinggi kepemakai. Sistem manajemen basisdata adalah perangkat lunak yang dirancang untuk membantu pengelolaan dan pemakaiaan koleksi data yang besar.

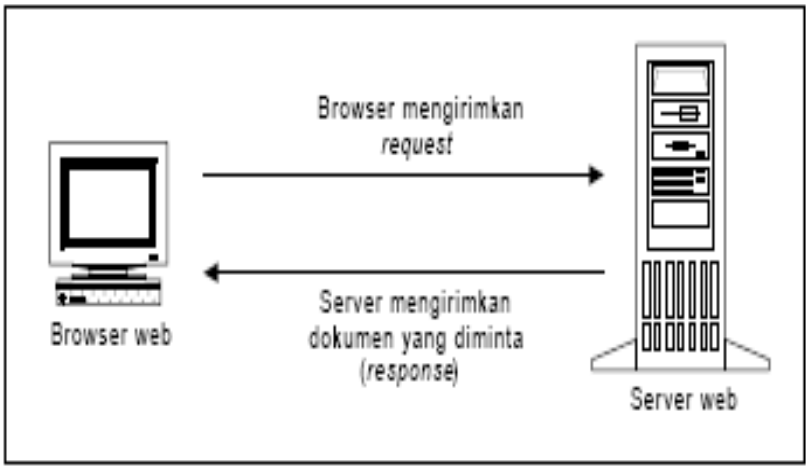
Pemakaian berurusan dengan abstrak tingkat tinggi yang lebih akrab, nyaman dan fleksibel bukan rincian fisik penyimpanan data.

2.7 Tinjauan Perangkat Lunak

Pada bagian ini penulis akan menguraikan penjelasan singkat mengenai perangkat lunak yang penulis gunakan dalam penyusunan laporan tugas akhir ini.

2.7.1 Web Server Apache

Server Apche adalah merupakan *engine server* yang mengelola semua interaksi dalam server dalam menangani komunikasi antara clien dengan server. Web *browser* berkomunikasi dengan *web server* lewat jaringan komunikasi menggunakan *protocol* HTTP. Browser mengirim pesan meminta dokumen atau layanan tertentu *web server*. Web server kemudian menanggapi dengan mengirim dokumen atau menjalankan layanan tertentu di *server* dan mengirim hasil menggunkan *protocol* HTTP. Kemudian browser akan menerima dokumen (HTML) tanggapan dari web server dan menampilkan dilayar. Protokol HTTP dirancang untuk dapat menggabung semua protocol internet seperti Gopher, Talnet, Wais, dan sebagianya dalam satu protokol tunggal. Semua lanyanan protocol lain dikemas sebagai layanan-layanan yang disediakan lewat interaksi web browser dan web server.



Gambar 2.1.1 web browser deng web server

2.7.2 PHP

PHP adalah teknologi yang diperkenalkan tahun 1994 oleh Rasmus Lerdorf. Beberapa versi awal yang dipublikasikan digunakan pada situs pribadinya untuk mencatat siapa saja yang mengakses daftar riwayat hidup *online*nya. Versi pertama digunakan oleh pihak lain pada awal tahun 1995 dan dikenal sebagai Personal *Home Page Tools*. Terkandung di dalamnya sebuah *parser engine* (mesin pengurai) yang sangat disederhanakan, yang hanya mampu mengelola macro khusus dan beberapa *utilities* yang sering digunakan pada pembuatan *home page*, seperti buku tamu, pencacah, dan hal semacamnya. *Parser* tersebut ditulis ulang pada pertengahan 1995 dan dinamakan PHP/FI version 2. FI (*Form Interpreter*) sendiri berasal dari kode lain yang ditulis juga oleh Rasmus, yang menterjemahkan HTML dari data. Ia menggabungkan *script* personal *home page* dengan *Form Interpreter* dan menambahkan dukungan terhadap *server database* yang menggunakan format mSQL, sehingga lahirlah PHP/FI tumbuh

dengan pesat, dan orang-orang mulai menyiapkan kode-kode programnya supaya bisa didukung oleh PHP.

PHP Singkatan dari ***Personal Home Page Tools,*** adalah sebuah bahasa scripting yang dibundel dengan HTML, yang dijalankan disisi *server*. Sebagian besar perintahnya berasal dari C, Java dan Perl dengan beberapa tambahan fungsi khusu PHP. Bahasa ini memungkinkan para pembuat aplikasi web menyajikan halaman HTML dinamis dan interaktif dengan cepat dan mudah, yang dihasilkan *server*, PHP juga dimaksudkan untuk mengganti teknologi lama seperti CGI (Common GateWay Interface), tetapi disini penulis tidak menerangkan tentang teknologi CGI.

2.7.3 MySQL

MySQL bukan lingkungan pengembangan aplikasi basisdata, tetapi satu DBMS kecil yang kompak. MySQL adalah DBMS kecil, kompak dan mudah digunakan MySQL ideal untuk aplikasi berukuran kecil dan menengah, namun telah menjanjikan untuk penggunaan besar. Implementasi MySQL beruoa *client/server* terdiri dari daemond mysqld (pada sistem Unix) dan beberapa program *client*. MySQL tersedia pada *platform* Unix, Windows NT/95/98. Pada *platform* Unix, MySQL menggunakan *multithreading* yang menaikkan kinerja dan skalabilitas secara bagus.

Adapun kelebihan dari MySQL untuk memilih MySQL dalam penggunaan database, yaitu :

a. Kecepatan
b. Kemudahan penggunaan

c. Ongkos

d. Dukungan bahasa query

e. Kapabilitas

f. Konektivitas dari keamanan

g. Portabilitas

h. Distribusi terbuka

Beberapa karakteristik MySQL

Tabel 2.1 karakteristik MySQL

No	Karakteristik	Deskripsi
1	Standart	MySQL mendukung *entry-level* ANSI SQL92 ODBc level 0-2
2	Character Set	MySQL secara *default* menggunakan ISO-8859-I (Latin I) charcter set untuk data pengurutan
3	Bahasa Pemrograman	MySQL mendukung pemrograman aplikasi dalam bahasa Java, Perl, PHP, dan sebagainya.
4	Tabel besar	MySQL menyimpan tiap relasi (tabel) pada file terpisah di direktori basisdata. Ukuran maksimum table dibatasi kemampuan sistem operasi menangani ukuran file.
5	Kecepatan, dan kemudahan pemakaian	MySQL kira-kira tiga sampai empat kali lebih cepat disbanding basisdata komersial, juga mudah dikelola.
6	MySQL	MySQL adalah *open-source relational* DBMS