BAB II

TINJAUAN PUSTAKA

II.1. Konsep Dasar Sistem Informasi

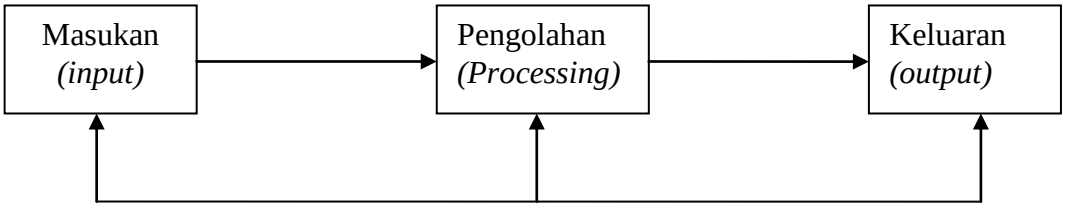
II.1.1. Sistem

Sistem adalah kumpulan dari elemen-elemen yang berinteraksi utuk mencapai suatu tujuan tertentu (Jogiyanto HM, 2005:2). Dengan demikian, secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur atau variabel-variabel yang saling terorganisasi, saling berinteraksi dan saling bergantung yang satu dengan yang lain (Prabowo Pudjo Widodo Herlawati, 2011:4).

Sistem diklasifikasikan sebagai sistem tertutup *(closed system)* dan sistem terbuka *(open system).* Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkunganluarnya. Sistem ini bekerja secara otomatis tanpa adanya turut campur tangan dari pihak diluarnya. Secara teoritis sistem tertutup ini ada, tetapi kenyataannya tidak ada sistem yang benar-benar tertutup, yang ada hanyalah *relatively closed system* (secara relatif tertutup, tidak benar-benar tertutup). Sistem terbuka dalah sistem yang berhubungan denga pengaruh dari linkungan luarnya. Sistem ini menerima masukan dan menghasilkan keluaran untuk lingkungan luar atau subsistem lainnya. Karena sistem sifatnya terbuka dan terpengaruh oleh lingkungan luarnya, maka suatu sistem harus mempunyai suatu sistem pengendalian yang baik. Sistem yang baik harus dirancang sedemikan rupa, sehingga secara relatif tertutup karena sistem

tertutup akan bekerja secara otomatis dan terbuka hanya untuk pegaruh yang baik saja (Jogiyanto HM, 2005:7)

Menurut scot (1996), sistm terdiri dari unsur-unsur seperti masukan (*input*), pengolahan (*proccessing*), serta keluaran (*output*). Ciri pokok sistem menurut Gapspert ada empat, yaitu sistem itu beroperasi dalam suatu lingkungan, terdiri atas unsur-unsur, ditandai dengan saling berhubungan, dan mempunyai satu fungsi dan tujuan yang utama (Prabowo Pudjo Widodo Herlawati, 2011 ; 7).



Gambar II.1. Model Sistem

(Sumber: Analisis dan Perancangan Sistem Informasi, Prabowo Pudjo Widodo Herlawati ,2011 ; 7)

II.1.2. Informasi

Informasi adalah data yang diolah menjadi bentuk yang lebih berarti bagi yang menerimanya. Sumber dari Informasi adalah data. Kualitas dari suatu informasi *(quality of information)* tergantung dari tiga hal, yaitu informasi harus akurat *(accurate)*, tepat pada waktunya *(timeliness)* dan *relevan.* Nilai dari informasi *(value of information)* ditentukan dari dua hal, yaitu manfaat dan biaya mendapatkannya. Suatu informasi dikatakan bernilai bila efektif dibandingkan dengan biaya mendapatkannya. Akan tetapi perlu diperhatikan bahwa informasi yang digunakan dalam suatu sistem umumnya digunakan untuk beberapa kegunaan. Sehingga tidak memungkinkan dan sulit untuk menghubungkan suatu bagian informasi pada masalah yang tertentu dengan biaya untuk memperolehnya, karena sebagian besar informasi dinikmati tidak hanya oleh satu pihak dalam

perusahaan. Lebih lanjut sebagian besar tidak dapat persis ditaksir keuntungannya dengan satuan nilai uang, tetapi dapat ditaksir nilai efektivitasnya. Pengukuran niali informasi biasanya dihubungkan dengan analisis *cost effectiveness* atau *cost benefit* (Jogiyanto HM, 2005:8-11).

II.1.3. Sistem Informasi

Sistem Informasi merupakan kombinasi teratur dari orang-orang, *hardware*, *software*, jaringan komunikasi, dan sumber daya data, yang mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi. Sistem Informasi juga adalah sekelompok elemen yang saling berhubungan atau berinteraksi hingga membentuk satu kesatuan (INDRAJANI, 2009:4)

John Bruch dan Gray Grudnitski mengemukakan bahwa sistem informasi terdiri dari komponen - komponen yang disebutnya dengan istilah blok bangunan (*building block*), yaitu blok masukan (*input block*), blok model (*block model*), blok keluaran (*output block*), blok teknologi (*technology block*), blok basis data (*database block*) dan blok kendali (*controls block*). Sebagai suatu sistem, keenam blok tersebut masing masing saling berinteraksi satu dengan yang lainnya memebentuk satu kesatuan untuk mencapai sasarannya (Jogiyanto HM 2005:12)

I.2. Pengolahan Data

Ada beberapa defenisi tentang data antara lain:

- Data adalah fakta atau observasi mentah yang biasanya mengenai fenomena fisik atau transaksi bisnis.

- Lebih khusus lagi, data adalah ukuran objektif dari atribut (karakteristik) dari entitas seperti orang, tempat, benda atau kejadian.
- Reprentasi fakta yang mewakili suatu objek seperti pelanggan, karyawan, mahasiswa dan lain-lain yang disimpan dalam bentuk angka, huruf, simbol, teks, gambar, bunyi dan kombinasinya.

  (INDRAJANI, 2009 : 2)

Data merupakan bentuk yang masih mentah yang belum dapat berceritera banyak, sehingga pelu diolah lebih lanjut. Data diolah melalui suatu model untuk dihasilkan informasi. Data yang diolah untuk menghasilkan informasi menggunakan suatu model proses yang tertentu. Misalnya data temperatur ruangan yang didapat adalah dalam satuan derajat fahrenheit dan data ini masih dalam bentuk yang kurang berarti bagi penerimanya yang terbiasa dengan satuan derajat celcius. Supaya dapat lebih berarti dan berguna dalam bentuk informasi, maka perlu diolah dengan melalui suatu model tertentu. Dalam hal ini dipergunakan model matematika yang berupa rumus konversi dari satuan derajat *fahrenheit* menjadi satuan *celcius* (Jogiyanto HM, 2009 : 9).

II.3. Konsep *Client-Server*

Client-Server adalah salah satu model komunikasi 2 komputer atau lebih yang berfungsi untuk melakukan pembagian tugas. *Client* bertugas untuk melakukan *input*, *update*, penghapusan, dan menampilkan data sebuah *database*. Sementara *server* bertugas menyediakan pelayanan untuk melakukan manajemen, yaitu menyimpan dan mengolah *database*.

Aplikasi *Client-Server* merupakan jawaban atas berkembangnya teknologi informasi, di mana sebuah perusahaan memiliki banyak departemen dan harus terhubung satu sama lain dalam melakukan akses data.

II.3.1. Arsitektur *Client-Server*

Terdapat beberapa arsitektur yang digunakan untuk melakukan pemrograman *database*, yaitu sebagai berikut :

1. Arsitektur *Standalone (1-Tier)*

   Konsep *1-Tier* (1- tingkat) adalah sebuah komputer yang mengakses sebuah database dari komputer sendiri. Dengan kata lain, aplikasi antarmuka *user* dan aplikasi DBMS terdapat pada komputer yang sama.

2. Arsitektur *Client-Server (2-Tier)*

   Arsitektur pada model ini membagi tugas antara komputer *client-server*. Komputer *client* bertugas menyediakan antarmuka untuk *user*, permintaan *(request)* data ke DBMS *Server,* serta pemrosesan data (mencakup logika penyajian data, logika pemrosesan data dan logika aturan bisnis). Komputer clien hanya mengirimkan sebuah *statement* untuk menambah *(insert)* data, mengubah *(update)* data, menghapus *(delete)* data dan terakhir meminta *(select)* data untuk ditampilkan melalui antarmuka yang telah dibuat oleh *programmer*. Pada sisi *server* model *2-Tier*, *server* bertanggungjawab terhadap penyimpanan, pengelolaan, melayani permintaan akses data dan pemrosesan oleh *client*.

3. Arsitektur *N-Tier*

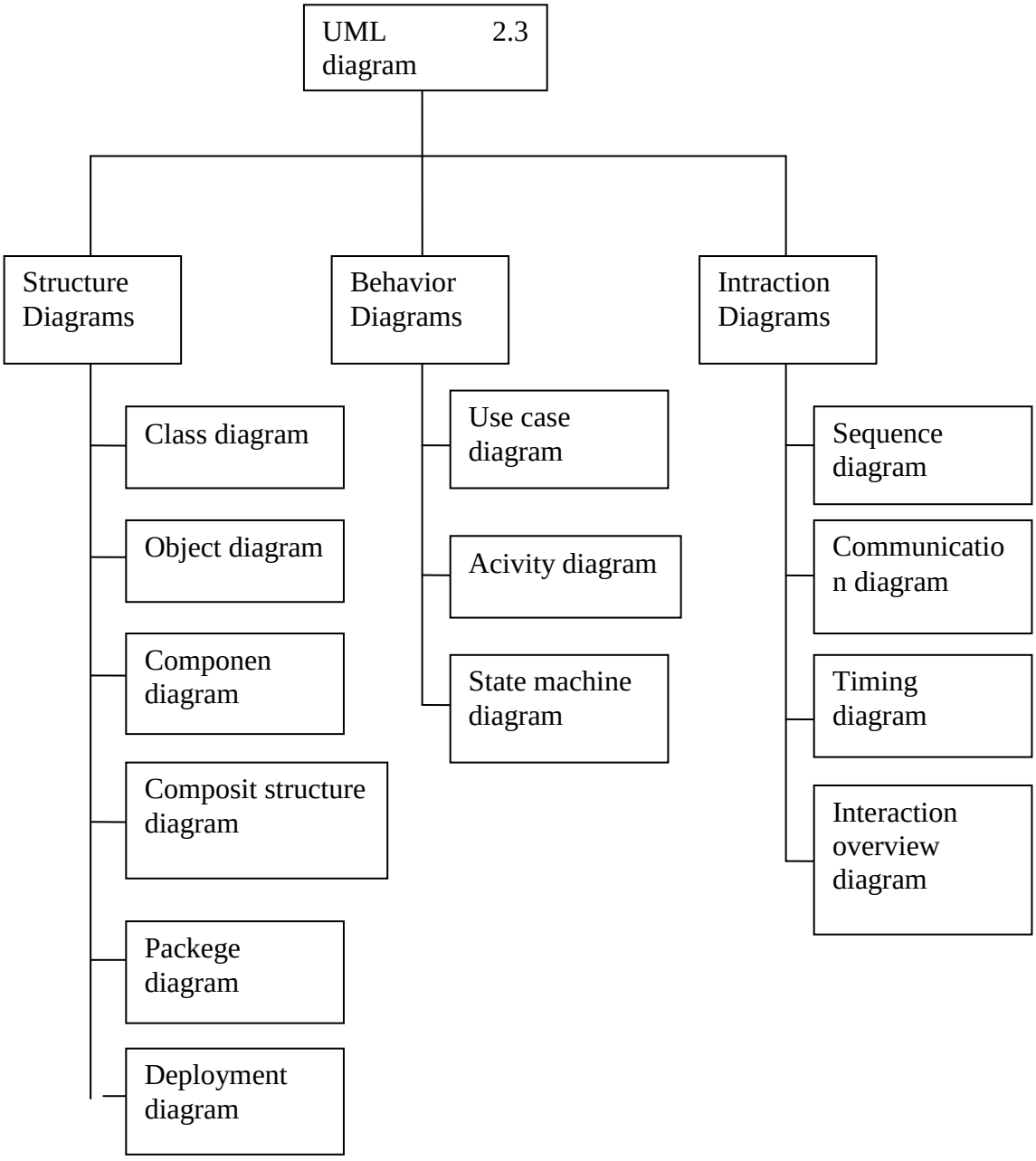
Arsitektur *n-tier* berarti membagi komponen menjadi n entitas, yaitu 1 *tier client* dan n-1 *tier server*. Seperti pada model sebelumnnya client bertugas mneyediakan antarmuka aplikasi, sedangkan *server* bertugas menyediakan data. Pada Model *n-tier* (sebagai contoh adalah *3-tier*), *server* dibagi menjadi 2, yaitu *server* yang dipakai sebagi *bussiness object (middle-tier)* dan satu *server* yang hanya menyimpan *database (server tier)*. Secara nyata model 3-*Tier* adalah pada jaringan internet yang memanfaatkan *database*.

(Wahana Komputer, 2010:5-9)

II.4. *Unified Modeling Language* (UML)

Pada perkembangan teknik pemrograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk memspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. UML hanya berfungsi untuk melakukan pemodelan (Rosa A.S-M.Shalahuddin : 2011).

Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Dapat dilihat pada gambar II.2. berikut :



Gambar: II.2. Diagram UML
(Sumber: Modul Pembelajaran Rekayasa Perangkat Lunak, Rosa A.S – M. Shalahuddin, 2011:121)

Penjelasan dari pembagian kategori:

a. *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

b. *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakukan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
c. *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi antar subsistem pada suatu sistem.
d. *Class diagram* yaitu diagram yang menggambarkan struktur sistem dari segi pendefinisiaan kelas-kelas yang akan dibuat membangun sistem.
e. *Object diagram* yaitu menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem.
f. *Component diagram* yaitu dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem.
g. *Composite structure diagram* yaitu menggambarkan struktur dari bagian-bagian yang saling terhubung maupun mendeskripsikan struktur pada saat sistem berjalan.
h. *Packege diagram* yaitu menyediakan cara mengumpulkan elemen-elemen yang saling terkait dalam diagram UML.
i. *Deployment diagram* yaitu menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi sistem
j. *Use case diagram* yaitu untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.
k. *Activity diagram* yaitu menggambarkan aliran kerja atau aktivitas dari sebuah sistem atau proses bisnis.

l. *State machine diagram* yaitu menggambarkan perubahan setatus atau transisi status dari sebuah mesin atau sistem.

m. *Sequence diagram* yaitu menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.

n. *Communication diagram* yaitu menggambarkan interaksi antar objek atau bagian dalam bentuk urutan pengiriman pesan.

o. *Timing diagram* yaitu mengambarkan tingkah laku sistem dalam periode waktu tertentu.

p. *Interaction overview diagram* yaitu mirip dengan diagram aktivitas yang berfungsi untuk menggambarkan sekumpulan urutan aktivitas. Diagram yang setiap titik mempresentasikan diagram interaksi.

(Rosa A.S – M. Shalahuddin, 2011:122)

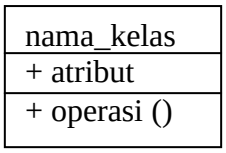
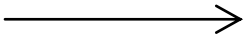
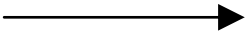
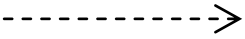
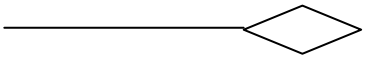
II.4.1. *Class Diagram* (Diagram Kelas)

Class Diagram atau Diagram Kelas menggambarkan struktur sistem dari segi pendefenisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

- Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas
- Operasi atau metode adalah fungsi-fungsi yang dimiliki suatu kelas.

Berikut adalah simbol-simbol yang ada pada diagram kelas, seperti ada Tabel II.1.:

Tabel II.1. Simbol-Simbol *Class Diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur system
Antarmuka / *interface*  nama_interface	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi/ *associate* 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah / *direct association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas lain, asosiasi biasanya juga disertai dengan *multiplicity*
generalisasi 	Relasi antar kelas dengan makna generelisasi-generalisasi- spesialisasi (umum-khusus)
Kebergantungan / dependency 	Relasi antar kelas dengan makna kebergantungan antar kelas
agresasi / *aggregation* 	Relasi antar kelas dengan makna semua bagian *(whole part)*

(Sumber: Modul Pembelajaran Rekayasa Perangkat Lunak, Rosa A.S – M. Shalahuddin, 2011:123)

II.4.2. *Diagram Use Case*

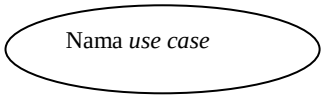
Diagram use case merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

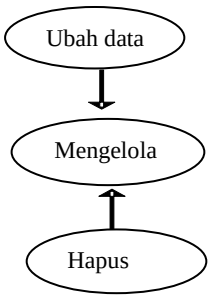
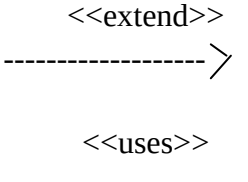
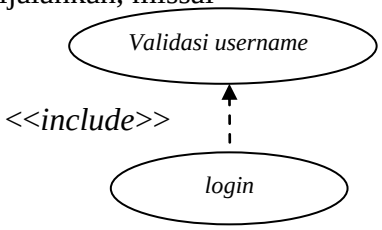
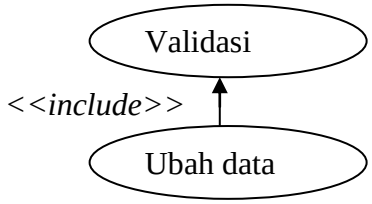
Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu:

a. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan di buat sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
b. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Berikut adalah tabel simbol-simbol diagram *Use Case* pada Tabel II.2. :

Tabel II.2. Simbol-Simbol *Diagram Use Case*

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja diawal frase nama *use case*
Aktor/ *actor*  Nama actor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat sendiri, jadi walupun simbol dari aktor adalah gambar orang, tapi belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Asosiasi/ *associate* _____	Komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case yang memiliki interaksi dengan aktor

Generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya:  Arah panah mengarah pada use case yang menjadi generalisasinya (umum)
Menggunakan /*include/ uses*  	Relasi use case tambahan kesebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini. Ada dua sudut pandang mengenai include di use case: 1. Include berarti use case yang ditambahkan akan selau dipanggil saat use case tambahan dijalankan, missal  2. *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah use case yang ditambah telah dijalankan sebelum *use case* tambahan dijalankan, misal: 

(Sumber: Modul Pembelajaran Rekayasa Perangkat Lunak, Rosa A.S – M. Shalahuddin, 2011:131-133)

II.4.3. *Diagram Activity*

Diagram aktivitas juga banyak digunakan untuk mendefinisikan seperti berikut:

a. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
b. Urutan atau pengelompokan tampilan dari sistem dimana setiap aktivitas dianggap memiliki sebuah rancangan tampilan antarmuka.
c. Rancangan pengujian setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan uji kasusnya.

Berikut simbol-simbol diagram aktivitas pada Tabel II.3. :

Tabel II.3. Simbol-Simbol Diagram Aktitas

Simbol	Deskripsi
setatus awal 	Sebuah diagram aktivitas memiliki sebuah status awal
aktivitas 	Aktivitas biasanya diawali dengan kata kerja
percabangan/ *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
penggabungan/ join 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Sebuah diagram aktivitas memiliki sebuah setatus akhir.

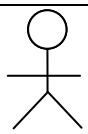
Swimlane Nama *swimlane*	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

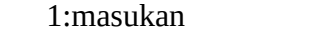
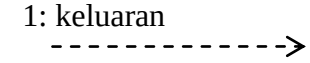
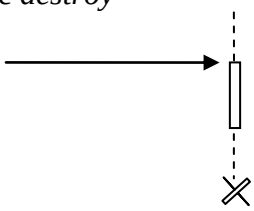
(Sumber: Modul Pembelajaran Rekayasa Perangkat Lunak, Rosa A.S – M. Shalahuddin, 2011:134-135)

II.4.3. *Diagram Sequence*

Banyaknya diagram sekuen yang harus digambarkan adalah sebanyak pendefinisian use casa yang memiliki proses sendiri atau yang penting semua use case yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak use case yang didefinisikan makan diagram sekuen yang harus dibuat juga semakin banyak. Berikut adalah symbol-simbol diagram sequence pada Tabel II.4. :

Tabel II.4. Simbol-simbol *Diagram Sequence*

Simbol	Deskripsi
nama aktor  nama aktor atau Nama aktor Tanpa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem, jadi waupun simbol dari aktor adalah gambar orang,tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Garis hidup/ *lifeline* 	Menyatakan kehidupan suatu objek

Objek Nama objek : nama kelas	Menyatakan objek yang berinteraksi pesan
waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe create 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri, arah panah mengarah pada objek yang memiliki opperasi /metode, karena ini memanggil operasi maka operasi yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi
Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim
Pesan tipe *return* 	Menyatan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembali
Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri objek yang lain, arah panah mengarah pada objek yang diakhiri, jika ada crate maka ada destroy.

(Sumber: Modul Pembelajaran Rekayasa Perangkat Lunak, Rosa A.S – M. Shalahuddin, 2011:138-139)

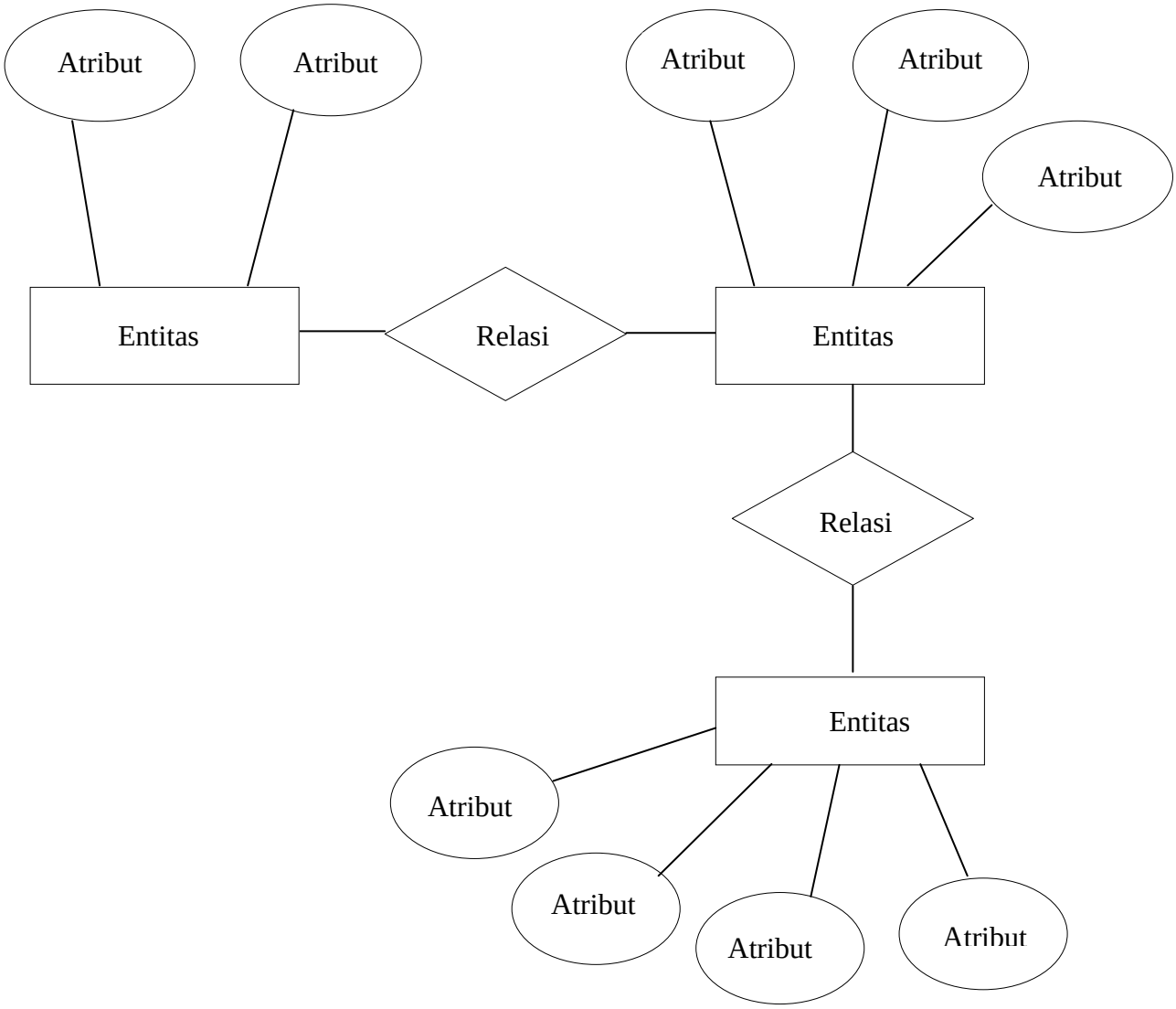
II.5. Diagram Relasi Entitas (*Entity Relationship Diagram*)

Entity Relatioship Model adalah suatu model data konseptual, yaitu memandang dunia nyata yang sepertinya terdiri dari entitas dan hubungan. Model secara visual menyajikan konsep tersebut dengan *Entity Relationship Diagram.* Untuk membangun dasar model *Entity Relationship* diperlukan entitas, hubungan, dan atribut. Kesatuan adalah konsep, baik riil atau abstrak, mengenai informasi yang dikumpulkan. Hubungan adalah asosiasi antar entitas. Atribut adalah sifat-sifat yang menguraikan kesatuan (Janner Simarmata, 2007:153).

Pemodelan awal basis data yang banyak digunakan adalah *Entity Relationship Diagram* (ERD). Pemodelan ini digunakan utuk pemodelan basis data relasional. Diagram ini menunjukkan hubungan antara entitas yang satu dengan yang lain dan juga bentuk hubungannya. Dengan adanya hubungan antar entitas ini maka seluruh data menjadi tergabung di dalam satu kesatuan yang terintegrasi (Dr.Ir.Eko Nugroho, M.Si, 2008:118).

Entity relationship diagram (ERD) mengilustrasikan struktur logis dari basis data, seperti gambar II.3. berikut:

Entity Relationship Diagram

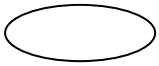
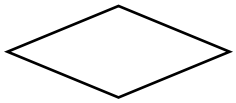


Gambar II.3. Entity Relationship Diagram

(Sumber : Perancangan Basis Data, Janner Simarmata, 2007:112)

Adapun notasi ataupun symbol yang digunakan dalam *Entity Relationship Diagram* (ERD) dapat dilihat pada Tabel II.3.berikut :

Tabel II.5. Notasi *Entity Relationship Diagram* (ERD)

Simbol	Nama	Keterangan
	*Entity*	Suatu entity merupakan suatu objek atau konsep mengenai tempat yang anda inginkan untuk menyimpan informasi.
	*Attribute*	Attributes adalah sifat-sifat atau karakteristik dari suatu entitas.
	*Relationship*	Mengilustrasikan bagaimana dua entitas berbagi informasi di dalam struktur basis data. Cara menggambar relasi adalah menghubungkan dua entitas terlebih dahulu, baru kemudian menge*drop* notasi relasi pada garis.
1 0..*	Asosiasi	Pengubung antar relasi dan entitas .

(Sumber : Perancangan Basis Data, Janner Simarmata, 2007 :112-113)

Didalam *Entity Relationship Diagram* (ERD) terdapat tingkatan hubungan antar entitas dilihat dari segi banyak atau tidaknya hubungan antar entitas tersebut yang disebut *mapping cardinalities*. Ada beberapa tingkatan hubungan yang mungkin terjadi, sebagai betikut :

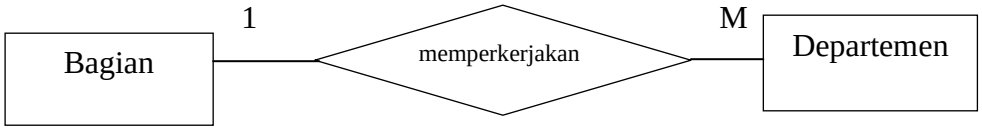
1. *One to One* (1:1)

   Bentuk hubungan yang menjelaskan satu entitas hanya berhubungan dengan satu entitas lain dan begitu pula sebaliknya. Contohya, pada pengajaran privat, satu guru satu siswa, seorang guru mengajar seorang siswa, seorang siswa diajar oleh seorang guru. Relasi *One to One* dapat dilihat pada Gambar II.4.

Gambar II.4. *One to One Relationship*
(Sumber : Basis Data, Janner Simarmata dan Iman Paryudi, 2006:64)

2. *One to Many*

Bentuk hubungan yang menjelaskan satu entitas dapat berhubungan dengan banyak entitas lainnya dan begitu pula sebaliknya. Contohnya, dalam suatu perusahaan satu bagian mempekerjan banyak pegawai, satu bagian memepekerjakan banyak pegawai, satu pegawai dalam satu bagian. Relasi *One to Many* dapat dilihat pada Gambar II.5.

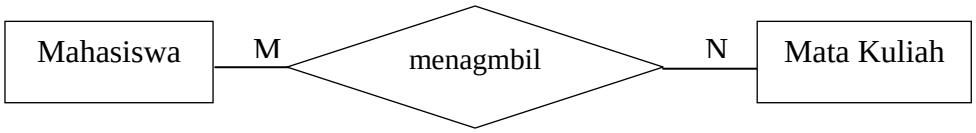


Gambar II.5. *One to Many Relationship*

(Sumber : Basis Data, Janner Simarmata dan Iman Paryudi, 2006:65)

3. *Many to Many*

Bentuk hubungan yang menjelaskan satu entitas dapat berhubungan dengan banyak entitas lainnya. Contohnya, dalam universitas, seorang mahasiswa dapat mengambil banyak mata kuliah, satu mahasiswa mengambil banyak mata kuliah dan satu mata kuliah diambil banyak mahasiswa. Relasi *Many to Many* dapat dilihat pada Gambar II.6. berikut:



Gambar II.6. *Many to Many Relationship*

(Sumber : Basis Data, Janner Simarmata dan Iman Paryudi, 2006:66)

II.6. Kamus Data (*Data Dictionary*)

Kamus Data (*Data Dictionary*) mencakup defenisi-defenisi dari data yang disimpan di dalm basis data dan diendalikan oleh sistem manajemen basis data. Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari defenisi

field, defenisi table, relasi table, dan hal-hal lainnya. Nama field data , jenis data(seperti teks atau angka atau tanggal), nilai-nilai yang valid untuk data, dan karakteristi-karakteristik lainnya yang akan disimpan dalam kamus dats. Perubahan-perubahan pada struktur datahanya dilakukan satu kali didalm kamus data: program-program aplikasi yang mempergunakan data tidak akan ikut berpenaruh. (Raymond Mcleod,Jr dan George P.Schell, 2008:171).

Dr.Ir.Eko Nugroho (2008:121) mendefenisikan bahwa, kamus data (*Data Dictionary*) adalah sebuah penjelasan tertulis lengkap dari data yang diisikan ke dalam *database*. Contoh kamus data adalah sebagai berikut pada Tabel II.6 :

Tabel II.6. Data Mahasiswa

No	Nama Item	Type Item	Lebar Item	Keterangan
1	No-Mhs	C	10	Nomor Mahasiswa
2	Nama	C	20	Nama mahasiswa lengkap
3	Alamat	C	20	Alamat lengkap mahasiswa
4	Tgl-lhr	D	8	Tanggal Lahir
5	Gol-drh	C	2	Golongan darah A, B, O,AB
6	Agama	C	1	Agama Mahasiswa 1:Islam 3:Budha 2:Kristiani 4:Hindu
7	Umur	N	2	Umur mahasiswa
8	Pek-Ortu	C	1	Pekerjaan Orang Tua 1: Pegawai negeri 2: wiraswasta
9	Jen-kel	C	1	Jenis Kelamin P: Pria W: Wanita
10	Alamat-kos	C	20	Alamat tempat tinggal saat ini

(Sumber : Sistem Informasi Manajemen, Dr.Ir.Eko Nugroho, 2008:122)

Keterangan dari Tabel II.6. :

1. *Nama Item* : adalah nama yang digunakan untuk menyebut/menamai item yang bersangkutan.

2. *Type Item* : adalah penjelas jenis item yang bersangkutan.
3. Lebar *Item* : yaitu lebar item data yang disediakan utuk menampung data yang akan direkam.
4. Keterangan : menjelaskan segala sesuatu informasi yang perlu diketahui berkaitan item data yang bersangkutan.

II.7. Normalisasi

Tujuan utama dalam pengembangan model *data logical* pada sistem basis data relational adalah menciptakan representasi suatu data secara akurat, hubungan antara data, dan batasan-batasannya. Normalisasi adalah suatu teknik dengan pendekatan *buttom-up* yang digunakan untuk membantu mengidentifikasi hubungan. Dimulai dari menguji hubungan *functional dependencies* antara atribut. Pengertuian lainnya adalah suatu tenik yang menghasilkan sekumpulan hubungan dengan sifat-sifat yang diinginkan dan memenuhi kebutuhan pada perusahaan. (INDRAJANI, 2009:109).

Terdapat empat bentuk normal yang biasa digunakan yaitu:

1. *First Normal Form (1NF)* atau Normalisasi Tingkat 1

   Merupakan sebuah relasi dimana setiap baris dan kolom berisikan satu dan hanya satu nilai.

   Contoh Bentuk 1NF :

N0. Penyewa	No.Properti	Alamat properti	Tgl Mulai Sewa	Tgl Akhir Sewa	Sewa Per Bulan	No.Pemilik	Nama Pemilik
S001	PR1	Jl. Kebon Jeruk No.1, Jakarta Barat	01/1/2008	01/12/2008	500.000	PP99	Matius
S001	PR4	Jl. Gatot Subroto No.100, Jakarta	01/1/2009	01/6/2009	1.000.000	PP77	Wahyu
S003	PR4	Jl. Gatot Subroto No.100, Jakarta	01/1/2008	01/12/2009	1.000.000	PP77	Wahyu

S003	PR1	Jl. Kebon Jeruk No.1, Jakarta Barat	01/1/2007	01/12/2007	500.000	PP99	Matius
S003	PR2	Jl. Sudirman No.2, Bandung	01/1/2008	01/12/2008	750.000	PP99	Matius

2. *Second Normal Form (2NF)* atau Normalisasi Tingkat 2

Sebuah relasi dalam 1NF dan stiap atribut *non-primary-key* bersifat *fully functionality depent* pada *primary key.* Contoh Bentuk 2NF :

Penyewa

No.Penyewa	Nama Penyewa
S001	Indrajani
S003	Indrayeti

SewaRumah

No. Penyewa	No.Prroperti	Tgl Mulai Sewa	Tgl Akhir Sewa
S001	PR1	01/1/2008	01/12/2008
S001	PR4	01/1/2009	01/6/2009
S003	PR4	01/1/2008	01/12/2009
S003	PR1	01/1/2007	01/12/2007
S003	PR2	01/1/2008	01/12/2008

Properti_Pemilik

No.Properti	Alamat properti	Sewa Per Bulan	No.Pemilik	Nama Pemilik
PR1	Jl. Kebon Jeruk No.1, Jakarta Barat	500.000	PP99	Matius
PR2	Jl. Sudirman No.2, Bandung	750.000	PP99	Matius
PR4	Jl. Gatot Subroto No.100, Jakarta	1.000.000	PP77	Wahyu

3. *Third Normal Form (3NF)* atau Normalisasi Tingkat 3

Sebuah relasi dalam 1NF dan 2NF dan dimana tidak terdapat atribut *non-primary-key* yang bersifat *transitively depent* pada *primary key*.

Contoh Bentuk 3NF :

Properti

No.Properti	Alamat property	Sewa Per Bulan	No.Pemilik
PR1	Jl. Kebon Jeruk No.1, Jakarta Barat	500.000	PP99
PR2	Jl. Sudirman No.2, Bandung	750.000	PP99
PR4	Jl. Gatot Subroto No.100, Jakarta	1.000.000	PP77

Pemilik

No.Pemilik	Nama Pemilik
PP99	Matius
PP77	Wahyu

4. *Boyce-Codd Normal Form (BCNF)*

   Berdasarkan pada functional dependencies yang dimasukkan ke dalam hitungan selurh candidat key dalm suatu relasi. BCNF juga memiliki batasan-batasan tambahan yang disamakan dengan defenisi umum dari 3NF.

5. *Four Normal Form (4NF)*

   Merupakan relasi dalam BCNF dan tidak memiliki nontrival multivalued depencies (MVD). MVD adalah reprentasi ketergantungan antara atribut-atribut (Misalnya A, B, C) dalam suatu relasi dimana setiap nilai A dalam himpunan nilai B dan C.

6. *Five Normal Form (5NF)*

   Merupakan suatu relasi yang tidak memiliki ketergantungan pada *join*.

   (Indrajani, 2009:120-121, 139-140)

II.8. Sistem Manajemen Basis Data (*Database Management System*)

Basis data adalah kumpulan informasi yang terorganisasi dan disajikan untuk mencapai tujuan khusus. Basis data terkomputerasi dapat di*update*, *file* bisa terorganisasi, informasinya dapat dibaca, dicari dengan cepat, dan bisa di*retrieve* menggunakan computer (Janner Simarmata, 2007:13).

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Kebutuhan basis data dalam sistem informasi meliputi:

1. Memasukkan, menyimpan, dan mengambil data
2. Membuat laporan berdasarkan data yang telah disimpan.

Menurut Janner Simarmata, (2007:14-15), sistem manajemen basis data (*Database Management System*) adalah suatu sistem perangkat lunak kompleks yang mengatur permintaan dan penyimpanan data ke dan dari *disk*. *Database Management System* menyediakan keamanan (*security*), privasi, integritas, dan *concurrency controls.* Secara umum, suatu sistem Manajemen Basis Data (DBMS) terdiri dari:

1. Suatu koleksi modul, program, dan tabel-tabel.
2. Suatu metode akses dan sebuah metodelogi akses.
3. Sekumpulana data, manipulasi, pelaporan, dan *tool-tool retrieval*.
4. Ketentuan *built in* untuk keamanan dan integritas data.
5. Sekumpulan *file, record,* serta uraian-uraian elemen.
6. Peraturan tentang logika untuk mengontruksi *file* dan menangani data.
7. Spesifikasi untuk menyimpan data fisik.

Pentingnya data bagi suatu organisasi atau perusahaan, maka hampir sebagian perusahaan memanfaatkan DBMS dalam mengelola data. Pengelolaan DBMS biasanya ditangani oleh tenaga ahli yang disebut *Database administrator*.

II.8.1. Kelemahan dan Kelebihan DBMS

Ada beberapa kelemahan dan kelebihan *Database Management System* (DBMS) yaitu sebagai berikut:

1) Kelemahan DBMS

- Harga DBMS mahal

- DBMS memerlukan banyak *software* pendukung yang mengakibatkan penambahan tempat penyimpanan dan memori.
- Pada DBMS terdapat pengaturan fungsi-fungsi sehingga DBMS menjadi *software* yang cuckup rumit dan kompleks.
- Penambahan biaya perangkat keras
- Adanya biaya konversi
- DBMS digunakan oleh banyak sistem informasi. Akibatnya mungkin beberapa pengguna sistem informasi akan merasakan hal-hal ynag tidak biasa sehingga kinerja akan lambat.
- Jika terjadi kerusakan pada DBMS, maka akan berdampak pada seluruh pengguna dan sistem informasi yang mengaskses DBMS.

2) Kelebihan DBMS

- Mengontrol redundansi data, dengan ada integrasi file ini maka berbagai duplikasi data dihilangkan.
- Jika ada perubahan yang terjadi dalam DBMS karena proses tambah, ubah, atau hapus data, maka pengguna-pengguna DBMS akan dapat mengakses nilai terbaru dalam DBMS secara cepat.
- Informasi yang lebih sejumlah data yang sama.
- Pemakaian data bersama.
- Meningkatnya integritas data.
- Meningkatnya keamanan data.
- Meningkatnya standarisasi.
- Meningkatnya skala ekonomi.

- Keseimbangan konflik kebutuhan.
- Meningkatnya akses data dan tanggapan.
- Meningkatnya produktivitas.
- Meningkatnya pemeliharaan.
- Meningkatnya konkurensi, meningkatnya *service backup* dan *recovery*.

II.8.2. Komponen *Database Management System*

Menurut INDRAJANI, (2009:12-17), komponen DBMS dibedakan menjadi 5 yaitu:

1. *Hardware* atau perangkat keras atau piranti keras.

   Hardware ini diperlukan oleh DBMS dan aplikasi. Contoh perangkat keras antara lain *personal computer, notebook, mainframe,* sampai sebuah jaringan komputer.

2. *Software,* perangkat lunak atau piranti lunak. Beberapa penggunaan *software* yaitu:

   - *Software* untuk sistem operasi komputer baik untuk PC, biasa ataupun *server*, contohnya *Windows XP, Windows 2000, Window NT, Windows 2003, Unix,* dan *Linux*.
   - *Software* untuk *database*, contohnya *Microsoft SQL 2000, Microsoft SQL 2005, Oracle, Mysql.*
   - *Software u*ntuk pemrograman, contohya di PC seperti *Java.Net 2005, Visual basic, C* dan *C++*. Di *mainframe* atau *AS400* contohnya *Cobol, RPG* dan *Fortran*.
   - *Software* untuk mengatur jaringan seperti *SISCO*.

3. Data, merupakan komponen terpenting DBMS karena data merupakan penghubung antara komputer dan manusia.
4. Prosedur, merupakan instruksi dan aturan yang menentukan perancangan dan penggunaan basis data di mana pengguna sistem dan pengelola basis data memerlukan dokumentasi ini untuk dan menggunakan sistem.
5. Orang, peranan orang ini dapat dibedakan menjadi beberapa bagian yaitu sebagai *Database administrator*, *Database desainer*, *programmer*, *end user*.

II.9. *Microsoft Visual Basic 2008*

Visual Basic 2008 merupakan aplikasi pemrograman yang menggunakan teknologi *.NET Framework*. Teknologi *.NET Framework* merupakan komponen *Windows* yang terintegrasi serta mendukung pembuatan, penggunaan aplikasi, dan halaman web. Teknologi *.NET Framework* mempunyai 2 (dua) komponen utama yaitu :

1. CLR (*Common Language Runtime*), digunakan untuk menjalankan aplikasi berbasis *.NET*
2. *Class Library*, adalah kelas pustaka atau perintah yang digunakan untuk membangun aplikasi.

Adapun persyaratan (*system Requirements*) yang harus dipenuhi Komputer agar bisa dijalankan dengan baik adalah seperti pada Tabel II.6.

Tabel II.7. *System Requirements Visual Basic* 2008

System	Syarat Minimal	Syarat yang direkomendasikan
Arsitektur	X86 dan X64 (WOW)	
Sistem Operasi	Microsoft Windows XP Service Packr 2, Microsoft Windows Server 2003, Windows Vista	
Prosesor	CPU 1.6 GHz (Giga Hertz)	Microsoft Windows XP dan Microsoft Windows Server 2003: CPU 2,2 GHz atau yang lebih tinggi. Window Vista: CPU 2,4 GHz
RAM	Windows XP dan Windows Server 2003: 384 MB(Mega Byte) , Window Vista 768 MB.	RAM 1024 MB/GB atau yang lebih besar.
Harddisk	Tanpa MSDN Ruang Kosong Harddisk pada drive penginstalan 2 GB. Sisa Ruang Harddisk kosong 1 GB. Dengan MSDN Ruang Kosong Harddisk pada drive penginstalan 3,8 GB (MSDN diinstal full) 2,8 GB untuk menginstall MSDN default. Kecepatan Harddisk 5400 RPM	Kecepatan harddisk 7200 RPM atau yang lebih tinggi
Display layar	1024 x 768 display	1280 x 1024 display

(Sumber: Wahana Komputer, Membuat Aplikasi *Client-Server* dengan *Visual Basic* 2008 2010 : 2)

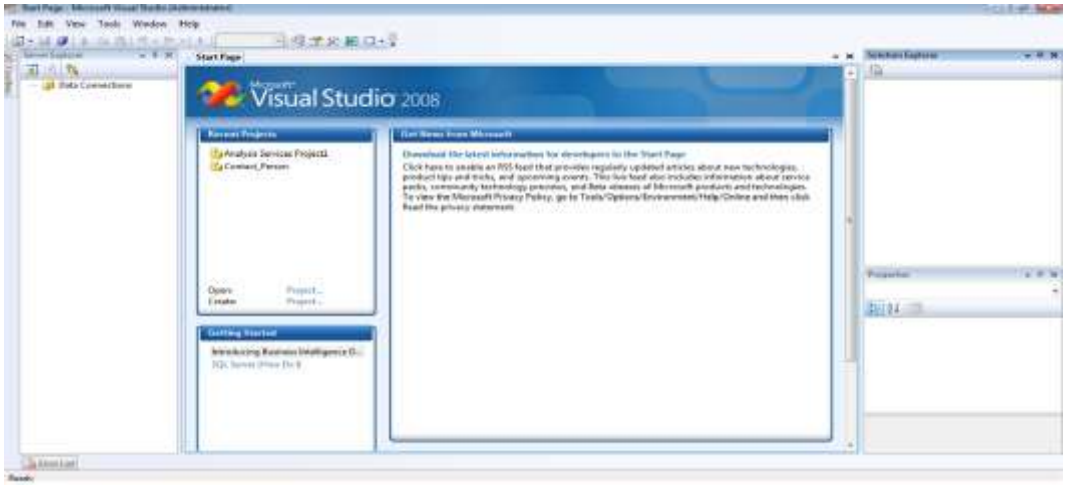
II.9.1. Mengenal Area Kerja *Visual Basic* 2008

Lingkungan kerja *Visual Basic* atau disebut *Integrated Development Environment* (IDE) adalah suatu lingkungan kerja tempat *programmer* melakukan pemrograman yang didukung oleh *compiler*, *editor* baik *editor* grafis maupun kode dan lain sebagainya untuk memudahkan pemrograman.

Adapun langkah membuka lingkungan kerja *Visual Basic* 2008 yaitu dengan cara, ikuti salah satu cara berikut untuk membuka *IDE Visual Basic*:

- Klik ganda shortcut Visual Studio 2008 pada desktop.
- Melalui Start > *Microsoft Visual Basic Studio 2008* > *Microsoft Visual Studio 2008.*

Saat menjalankan *Visual Basic 2008* pertama sekali yang muncul adalah kotak dialog, dapat dilihat pada Gambar II.8. berikut :



Gambar II.7. Area Kerja Visual studio saat pertama kali dijalankan (Sumber : Wahana Komputer, 2010:2)

II.10. *MySQL Server*

MySQL adalah salah satu *software* satu sistem manajemen *database* (DBMS) *Structured Query language* (SQL) yang bersifat *open source*. *SQL* adalah bahasa standar untuk mengakses *database* dan didefenisikan dengan standar ANSI/ISO SQL. *MySQL* dikembangakan , disebarluaskan dan didukung oleh *MySQL AB*. *MySQL AB* adalah perusahaan komersial yang didirian oleh

pengembang *MySQL*. *MySQL* merupakan aplikasi *Relational Database Management System (RDBMS)* yang dapat digunakan sebagai aplikasi *client-server* atau sistem *embedded*. *MySQL* mempunyai beberapa sifat yang menjadikannya sebagai salah satu *software database* yang banyak digunakan oleh pemakai di seluruh dunia. Sifat-sifat yang dimiliki oleh *MySQL* antara lain :

- *MySQL* merupakan DBMS (*Database Management System*)
- *Database* adalah kumpulan data yang terstruktur. Data dapat berupa daftar belanja, kumpulan gambar atau lebih luas yaitu informasi jaringan perusahaan. Agar dapat menambah, mengakses, dan memproses data tersimpan pada sebuah komputer *database*, kita membutuhkan sistem manajemen *database* (DBMS) seperti *MySQL Server*. Sejak komputer sangat baik dalam menangani sejumlah besar data, sistem manajemen *database* (DBMS) memainkan peran utama dalam perhitungan baik sebagai peralatan yang berdiri sendiri maupun bagian sebuah aplikasi
- *MySQL* merupakan RDBMS (*Relational Database management System*)
- *Database* relasional menyimpan data pada tabel-tabel yang terpisah, bukan menyimpan data dalam ruang penyimpanan yang besar. Hal ini menambah kecepatan dan fleksibilitas.
- *MySQL* merupakan *software open source.*
- *Open source* berarti setiap orang dapat menggunakan dan mengubah *software* yang bersangkutan. Setiap orang dapat men-*download software MySQL* dari internet dan menggunakanna tanpa bayar. Bahkan jika menghendakinya, anda bisa mempelajari kode sumber dan mengubahna

sesuai yang anda butuhkan. *Software MySQL* menggunakan GNU/GPL (*General Public License*).

- *MySQL* mempunyai performa yang sangat cepat, dapat dipercaya dan mudah digunakan.
- *MySQL Server* sebenarnya dikembangkan untuk menangani *database* besar lebih cepat daripada solusi yang ada dan telah berhasil digunakan pada lingkungan produksi dengan permintaan tinggi untuk beberapa tahhun terakhir. Walaupun dibawah penegmbang yang sama, *MySQL Server* sekarang menawarkan kumpulan fungsi yang banyak dan bermanfaat. Konekvitas, kecepatan dan keamanan yanag dimiliki *MySQL Server* membuatnya sangat cocok untuk mengakses *database internet*.
- *MySQL Server* bekerja pada *client-server* atau pada sistem *embedded*.
- *Software MySQL Server* adalah sistem *client-server* yang terdiri atas *multi-threaded SQL Server* yang mendukung backend berbeda, beberapa program *client* dan pustaka (*libraries*) berbeda, peralatan administrasi, dan jangkauan luas *API* (*Application Programming Interfaces*)
- Ada pula *MySQL Server* sebagai pustaka *embedded multi-threaded* yang dapat menhubungkan kita ke dalm aplikasi untuk mendapatkan *MySQL Server* lebih kecil, lebih cepat, dan lebih mudah untuk mengatur produk *standalone*.
- *MySQL* mempunyai sejumlah besar *software* pendukung.
- Aplikasi atau bahasa kesukaan anda sangat mungkin mendukung *database* MySQL *Server*.

(Wahana Komputer, 2010:26-27)

MySQL Server mendukung banyak tipe data yang dapat disimpan pada sebuah kolom. Terdapat tiga kategori tipe data yang didukung olehn *MySQL Server*, yaitu tipe data numerik, *string*, serta penanggalan dan waktu. Sebuah data yang akan disimpan harus sesuai dengan kolom yang bersangkutan. Dengan mengetahui tipe data, anda akan mengetahui cara menentukan tipe data yang sesuai untuk sebuah kolom pada tabel. Hal ini sangat penting untuk meningkatkan performa *database* (Wahana Komputer, 2010:30)

MySQL merupakan *DBMS* yang mengacu kepada *statement SQL (Structured Query Language), statement SQL* yang digunakan umumnya dibagi menjadi 3 subbahasa yaitu sebagai berikut:

1. *Data Defenition Language* (DDL)

   Data Defenition Language (DDL) adalah kumpulan perintah yang digunak untuk membangun databaseatau dengan kata lain, suatu bentuk bahasa yang digunakan untuk mendefenisikan struktur *table*. Jadi dengan bahasa ini anda akan mampu menciptakan *database* dan *table*. Didalam DDL, *SQL* mempunyai empat *statement* yang digunakan untuk mendefenisikan *database* dan table, yaitu *Create, Alter, Drop* dan *Rename.* (Wahana Komputer, 2010:34)

2. *Data Manipulation Language* (DML)

   Data Manipulation Language (DML) adalah kumpulan *statement* atau perintah *SQL* yang digunakan untuk mengelola data pada suatu *database*. Terdapat beberapa perintah yang dikategorikan sebagai DML yaitu

INSERT, SELECT, UPDATE, DELETE, TRUNCATE, DO, REPLACE, HANDLER, dan *LOAD DATA INFILE*. Perintah-perintah tersebut berhubungan dengan data pada *table*, sehingga perintah DML hanya digunakan setelah anda menggunakan perintah DDL (Wahana Komputer, 2010:45)

3. *Data Control Language* (DCL)

   Data Control Language (DCL) atau *Database Administration Statement* adalah kumpulan perintah yang digunakan untuk melakukan manajemen tentang pemakai dan *table* yang akan bertujuan utuk menjaga keamanan data pada *server*. Perintah pada pada DCL yaitu *Create User, Drop User, Rename User, Grant, Revoke,* dan *Set Password, Show.*

   (Wahana Komputer, 2010:54)