BAB II

TINJAUAN PUSTAKA

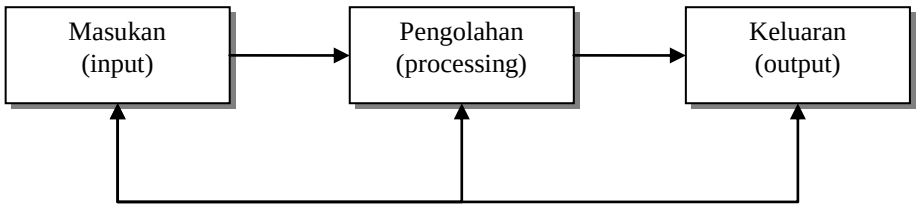
II.1. Konsep Dasar Sistem

Konsep dasar sistem akan menguraikan beberapa pengertian sistem, karakteristik sistem, pengertian dan komponen sistem informasi.

II.1.1. Pengertian Sistem

Secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur atau variabel-variabel yang saling terorganisasi, saling berinteraksi, dan saling bergantung sama lain. Murdick dan Ross (1993) mendefinisikan sistem sebagai perangkat elemen yang digabungkan satu sama lainnya untuk suatu tujuan bersama.

Menurut *Scott (1996)*, sistem terdiri dari unsur-unsur seperti masukan (*input*), pengolahan (*processing*), 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 atau tujuan utama.

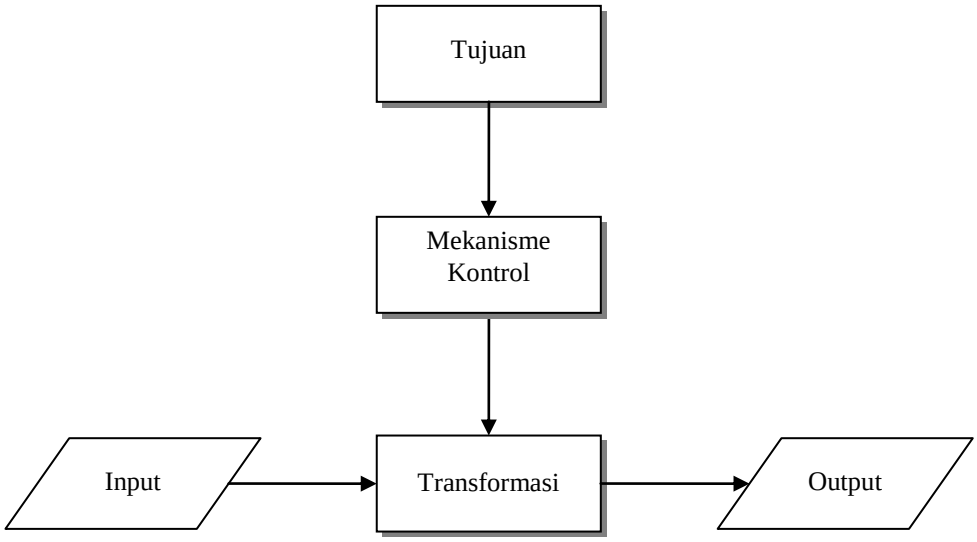


Gambar II.1. Model Sistem

Sumber : Hanif Al Fatta (2007:4)

Gambar di atas menunjukkan bahwa sistem atau pendekatan sistem minimal harus mempunyai empat komponen, yakni masukan, pengolahan, keluaran, dan balikan atau *control*.

Sementara *Mc. Leod (1995)* mendefinisikan sistem sebagai sekelompok elemen-elemen yang terintegrasi dengan maksud yang sama untuk mencapai suatu tujuan. Sumber daya mengalir dari elemen output dan untuk menjamin prosesnya berjalan dengan baik maka dihubungkan dengan mekanisme *control*. Untuk lebih jelasnya elemen sistem tersebut dapat digambarkan dengan model sebagai berikut:



Gambar II.2. Model Hubungan Elemen-Elemen Sistem

Sumber : Hanif Al Fatta (2007:4)

Banyak ahli mengajukan konsep sistem dengan deskripsi yang berbeda, tetapi pada prinsipnya hampir sama dengan konsep dasar sistem umumnya. *Schronderberg (1971)* dalam *Suradinata (1996)* secara ringkas menjelaskan bahwa sistem adalah :

1. Komponen-komponen yang saling berhubungan satu sama lain.
2. Suatu keseluruhan tanpa memisahkan komponen pembentuknya.
3. Bersama-sama dalam mencapai tujuan.
4. Memiliki input dan output yang dibutuhkan oleh sistem lainnya.
5. Terdapat proses yang mengubah input menjadi output.
6. Menunjukkan adanya entropi.
7. Memiliki aturan.
8. Memiliki subsistem yang lebih kecil.
9. Memiliki deferensi antar subsistem.
10. Memiliki tujuan yang sama meskipun mulainya berbeda. (Hanif Al Fattah, 2007 : 5)

II.1.2. Karakteristik Sistem

Untuk memahami atau mengembangkan suatu sistem, maka perlu membedakan unsur-unsur dari sistem yang membentuknya. Berikut adalah karakteristik sistem yang dapat membedakan suatu sistem dengan sistem yang lainnya :

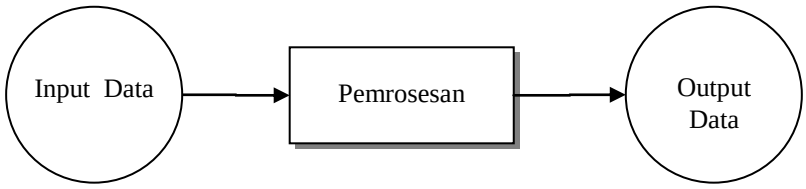
1. Batasan (*boundary*) : Penggambaran dari suatu elemen atau unsur mana yang termasuk di dalam sistem dan mana yang di luar sistem.
2. Lingkungan (*environment*) : Segala sesuatu di luar sistem, lingkungan yang menyediakan asumsi, kendala, dan input terhadap suatu sistem.
3. Masukan (*input*) : Sumber daya (data, bahan baku, peralatan, energi) dari lingkungan yang dikonsumsi dan dimanipulasi oleh suatu sistem.

4. Keluaran (*output*) : sumber daya atau produk (informasi, laporan, dokumen, tampilan layer computer, barang jadi) yang disediakan untuk lingkungan sistem oleh kegiatan dalam suatu sistem.
5. Komponen (*component*) : Kegiatan-kegiatan atau proses dalam suatu sistem yang mentransformasikan input menjadi bentuk setengah jadi (*output*). Komponen ini bisa merupakan subsistem dari sebuah sistem.
6. Penghubung (*interface*) : Tempat di mana komponen atau sistem dan lingkungannya bertemu atau berinteraksi.
7. Penyimpanan (*storage*) : Area yang dikuasai dan digunakan untuk penyimpanan sementara dan tetap dari informasi, energi, bahan baku, dan sebagainya. Penyimpanan merupakan suatu media penyangga di antara komponen tersebut bekerja dengan berbagai tingkatan yang ada dan memungkinkan komponen yang berbeda dari berbagai data yang sama. (Hanif Al Fattah, 2007 : 5-6)

II.1.3. Pengertian Sistem Inforamsi

Untuk memahami pengertian sistem informasi, harus dilihat keterkaitan antara data dan informasi sebagai entitas penting pembentuk sistem informasi. Data merupakan nilai, keadaan, atau sifat yang berdiri sendiri lepas dari konteks apapun. Sementara informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau mendatang (*Davis, 1995*). *Mc Leod (1995)* mengatakan bahwa informasi adalah data yang telah diproses, atau data yang memiliki arti.

Akhirnya Sistem Informasi Manajemen (SIM) dapat didefinisikan sebagai suatu alat untuk menyajikan informasi dengan cara sedemikian rupa sehingga bermanfaat bagi penerimanya (*Kertahadi, 1995*). Tujuannya adalah untuk menyajikan informasi guna pengambilan keputusan pada perencanaan, pemrakarsaan, pengorganisasian, pengendalian kegiatan operasi subsistem suatu perusahaan, dan menyajikan sinergi organisasi pada proses (*Murdick dan Ross, 1993*). Dengan demikian, sistem informasi berdasarkan konsep (*input, processing, output – IPO*) dapat dilihat pada gambar II.3.



Gambar II.3. Konsep sistem informasi

Sumber : Hanif Al Fatta (2007: 9)

II.1.4. Komponen Sistem Informasi

Stair (1992) menjelaskan bahwa sistem informasi berbasis komputer (CBIS) dalam suatu organisasi terdiri dari komponen-komponen berikut :

1. Perangkat keras, yaitu perangkat keras komponen untuk melengkapi kegiatan memasukkan data, memproses data, dan keluaran data.
2. Perangkat lunak, yaitu program dan instruksi yang diberikan ke komputer.
3. *Database*, yaitu kumpulan data dan informasi yang diorganisasikan sedemikian rupa sehingga mudah diakses pengguna sistem informasi.

4. Telekomunikasi, yaitu komunikasi yang menghubungkan antara pengguna sistem dengan sistem komputer secara bersama-sama ke dalam suatu jaringan kerja yang efektif.
5. Manusia, yaitu personel dari sistem informasi, meliputi manajer, analis, programmer, dan operator, serta yang bertanggung jawab terhadap perawatan sistem (Hanif Al Fattah, 2007 : 10)

Prosedur, yakni tata cara yang meliputi strategi, kebijakan, metode, dan peraturan-peraturan dalam menggunakan sistem informasi berbasis komputer

Sementara *Burch dan Grudnitski (1986)* berpendapat, sistem informasi yang terdiri dari komponen-komponen di atas disebut dengan istilah blok bangunan (*building block*), yaitu blok masukan (*input block*), blok model (*model block*), blok keluaran (*output block*), blok teknologi (*technology block*), dan blok kendali (*control block*). Sebagai suatu sistem, keenam blok tersebut masing-masing saling berinteraksi satu dengan yang lainnya membentuk satu kesatuan untuk mencapai sasarannya.

1. Blok Masukan. Input mewakili data yang masuk ke dalam sistem informasi. Input di sini termasuk metode-metode dan media untuk menangkap data yang akan dimasukkan yang dapat berupa dokumen-dokumen dasar.
2. Blok Model. Blok ini terdiri dari kombinasi prosedur, logika, dan model matematika yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara tertentu untuk menghasilkan keluaran yang diinginkan.

3. Blok Keluaran. Produk dari sistem informasi adalah keluaran yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkat manajemen serta semua pemakai sistem.
4. Blok Teknologi. Teknologi merupakan kotak alat (*tool box*) dalam sistem informasi. Teknologi digunakan untuk menerima input, menjalankan model, menyimpan dan mengakses data, menghasilkan sekaligus mengirimkan keluaran dan membantu pengendalian dari sistem secara keseluruhan.
5. Blok Database. Database merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya.
6. Blok Kendali. Pengendalian perlu dirancang dan diterapkan untuk meyakinkan bahwa hal-hal yang dapat merusak sistem dapat dicegah ataupun bila terlanjur terjadi kesalahan-kesalahan dapat langsung cepat diatasi.

Sementara, menurut pendapat *Davis (1995)*, sistem informasi manajemen terdiri dari elemen-elemen berikut :

1. Perangkat keras komputer (*hardware*)
2. Perangkat lunak (*software*), yang terdiri dari perangkat lunak sistem umum, perangkat lunak terapan, dan program aplikasi.
3. Database
4. Prosedur
5. Petugas operasional. (Hanif Al Fattah, 2007 : 10 - 11).

II.2. Penggajian

Gaji adalah sebuah komponen yang mutlak dikeluarkan oleh perusahaan sebagai kompensasi bagi karyawan, yang mana hal ini untuk menjamin keberlangsungan perusahaan itu sendiri, bayangkan anda bekerja tetapi tidak digaji ? Apa tidak segera kabur.

secara konsep ada berbagai macam pendekatan, misalnya *pay for position, pay for person, pay for performance, pay for competence, equal job equal pay, skill based pay,* dan *merit based pay*.

Pay for position dimana pegawai dihargai berdasarkan posisi atau jabatannya dalam perusahaan. Perusahaan menerapkan sistem ini untuk jenis pegawai dengan pekerjaan struktural dan dikarenakan sistem ini paling mudah dilakukan kebanyakan perusahaan menerapkan sistem ini dalam penggajian pegawai mereka.

Pay for person yaitu karyawan dihargai berdasarkan keahlian atau kompetensi yang dimiliki, sistem ini diberlakukan bagi pegawai fungsional, dimana pegawai dihargai atas keahlian atau kompetensi yang bresifat teknis atau khusus.

Pay for performance dimana karyawan dihargai berdasarkan kinerjanya pada suatu periode tertentu. Kinerja setiap pegawai diukur sesuai dengan pencapaian indikator kinerja yang telah ditetapkan targetnya.

Equal job equal pay adalah upah yang sama untuk jenis pekerjaan yang sama. Jika ada dua orang atau lebih mengerjakan pekerjaan yang sama maka upah mereka mesti sama.

Competetence pay adalah penggajian yang dihubungkan secara langsung dengan bukti langsung objektif terjadinya peningkatan dalam pengetahuan teknis, keterampilan dan keahlian seorang karyawan.

Skill based pay yaitu pembayaran dimana para pekerja digaji berdasarkan pengetahuan dan keterampilannya dari pada posisinya di perusahaan.

Merit pay (berdasarkan jasa) yaitu sistem penggajian dimana pekerja digaji berdasarkan *performancenya,* pencapaian financial pekerja berdasarkan pada hasil yang dicapai oleh individu itu sendiri.

Kebijakan dalam menentukan gaji biasanya dipegang oleh bagian HRD (*Humen Resource Development*) atau harus biasanya dikenal dengan manajemen personalia, kebijakan yang dibuat harus menciptakan suatu kepastian hukum, rasa aman dan mencegah timbulnya perselisihan. Artinya setiap kebijakan yang diterapkan harus memenuhi rasa keadilan, jelas, transparan, diterapkan konsisten tanpa memihak dan mendapat komitmen dari dukungan penuh dari pihak manajemen tertinggi dalam implementasinya.

Untuk bisa memenuhi rasa keadilan bisa diwujudkan dengan penerapan teori *equal job equal pay*. Kebijakan penggajian wajib dibuat secara formal dan tertulis, diketahui dan dipahami dengan benar oleh seluruh karyawan kecuali nilai rupiahnya, tetapi setiap karyawan wajib mengetahui standar nilai rupiah yang diterapkan bagi dirinya sendiri (asas transparan dan jelas).

Kebijakan juga bersifat konsisten, tidak memihak, selama seseorang adalah karyawan di suatu perusahaan maka wajib mematuhi dan taat pada aturan kebijakan yang diberlakukan.

Yang paling penting lagi, rencana penerapan sistem penggajian juga didukung oleh komitmen Top Manajemen, hal yang sangat penting agar tercipta respek dari seluruh karyawan terhadap kebijakan yang diberlakukan. Top Manajemen juga wajib mendukung dan taat pada kebijakan yang diberlakukan (Miftakhul Huda, dkk; 2013 : 1-2).

II.2.1. Sistem Informasi Akuntansi

Organisasi tergantung pada sistem informasi untuk dapat berdaya saing. Informasi juga menerapkan sumber daya, sama seperti pabrik dan peralatan. Produktivitas, sebagai faktor yang penting untuk mempertahankan daya saing perusahaan, dapat ditingkatkan dengan sistem informasi yang lebih baik. Akuntansi, sebagai suatu sistem informasi, mengidentifikasi, mengumpulkan, memproses, dan mengkomunikasikan informasi ekonomi mengenai suatu entitas ke berbagai kelompok orang. Informasi merupakan suatu datayang diorganisasi yang dapat mendukung ketepatan pengambil keputusan. Sistem merupakan sekumpulan sumber daya yang saling terkait untuk mencapai suatu tujuan.

Sistem Informasi Akuntansi (SIA) adalah merupakan kumpulan sumber daya, seperti manusia dan peralatan yang dirancang untuk mengubah data keuangan dan data lainnya ke dalam informasi. Informasi tersebut dikomunikasikan kepada para pembuat keputusan, Sistem informasi akuntansi melakuka hal tersebut entah dengan sistem manual atau melalui sistem terkomputerisasi (George H. Bodnar, dkk; 2006 :3).

II.3. Basis Data (*Database*)

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 dari basis data dan menyajikan dalam bentuk yang bisa dimenegrti. 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.

Basis data biasanya memiliki dua bagian utama. Pertama, file yang memegang basis data fisik. kedua, perangkat lunak sistem manajemen basis data (DBMS) menggunakan aplikasi untuk mengakses data. DBMS bertanggung jawab menguatkan struktur basis data termasuk :

a. Memelihara hubungan antar data di dalam basis data.
b. Memastikan bahwa data tersimpan secara tepat, dan menetapkan aturan hubungan data agar tidak terlanggar.\
c. Pemulihan (*recovery*) semua data dari kegagalan sistem (Janner Simarmata; 2007 : 2).

II.3.1. Normalisasi

Redudansi data cenderung melebihi ukuran dari basis data dan itu menjadi sebuah masalah yang sangat serius dalam media basis data yang besar. Selain itu, redudansi data bisa mendorong kea rah yang tidak normal (*anomalies*). Untuk memahami permasalahan yang bisa mengakibatkan pemborosan (*redudansi*), kita harus mengetahui pengertian redudansi lebih dekat lagi.

Sebelumnya, mari kita mulai dengan memgamati bahwa atribut dari skema tabel bisa digolongkan ke dalam tiga kelompok :

1. Atribut yang digunakan untuk tujuan identifikasi.
2. Atribut yang digunakan untuk tujuan informasi
3. Atribut yang digunakan untuk tujuan keduanya, baik identifikasi maupun informasi.

a. Bentuk Tidak Normal.

untuk membuat perancangan basis data, Anda pasti sudah mengenal sejumlah bentuk khusus, sifat-sifat, atau batasan skema tabel yang dimiliki untuk mencapai suatu tujuan yang diinginkan, misalnya memperkecil redudansi. Bentuk itu disebut “Bentuk Normal”

ada enam bentuk normal yang dikenal yaitu :

1. Bentuk Normal Pertama (1NF)
2. Bentuk Normal Kedua (2NF)
3. Bentuk Normal Ketiga (3NF)
4. Bentuk Normal Boyce Codd (BCNF)
5. Bentuk Normal Keempat (4NF)
6. Bentuk Normal Kelima (5NF)

Bentuk normal pertama sampai ketiga (dibuat oleh E.F. Codd) merupakan bentuk normal yang umum dipakai. Maksudnya, pada kebanyakan relasi, persoalan anomaly tidak akan muncul lagi bila ketiga bentuk normal tersebut telah terpenuhi.

b. Bentuk Normal Pertama (1 NF)

Bentuk normal pertama sangat sederhana. skema tabel disebut dalam bentuk normal pertama jika nilai atribut tidak terpisahkan. Untuk mengilustrasikannya, semua penulis buku di dalam atribut tunggal disebut penulis.

Berikut sebuah contoh :

ISBN = 979 -763-120-6

Judul = Basis Data

Penulis = Janner Simarmata Dan Iman Prayudi

Penerbit = Andi Yogyakarta

Oleh karena itu skema dalam kasus ini mengizinkan lebih dari satu nama penulis sebagai atribut penulis, maka skema bukan dalam bentuk normal pertama. Tentu saja, salah satu dari permasalahan yang jelas nyata dengan atribut penerbit adalah ketidakmungkinan mengurutkan data menggunakan nama penulis individu. Akan lebih sulit juga, sebagai contoh, bila menyiapkan label surat untuk setiap penulis dan seterusnya.

c. Bentuk Normal Kedua (2 NF)

Berdasarkan skema tabel T memiliki bentuk normal kedua jika semua atribut informasi (atribut yang tidak memiliki kunci mana pun) adalah atribut dari entitas lain di dalam skema tabel dan bukan dari kelas entitas lainnya. Dengan kata lain, atribut informasi menyediakan informasi secara rinci tentang entitas di dalam kelas entitas itu dan bukan beberapa entitas lain.

d. Bentuk Normal Ketiga (3 NF)

Bentuk normal kedua adalah baik, tetapi kita bisa melakukan yang lebih baik lagi. Kita sudah melihat bahwa jika skema tabel berbentuk normal kedua, maka bukan informasinya yang kuat, atribut tergantung pada subset yang sesuai dengan kunci. Bagaimanapun juga, ada kemungkinan lain yang tidak diinginkan. Lihat contoh berikut :

Asumsi dan skema tabel berikut adalah untuk tujuan ilustrasi, yakni bahwa tidak ada dua buku dengan judul yang sama dengan penerbit yang sama :

{Judul, KdPenerbit, JmlHalaman, Harga}

Satu-satunya kunci untuk skema tabel itu adalah **{Judul, KdPenerbit, JmlHalaman,** dan **Harga** hanya lah atribut informasi.

e. Bentuk Normal Boyce Codd (BCNF)

BCNF merupakan bentuk normal sebagai perbaikan terhadap 3 NF. Suatu relasi BCNF selalu memenuhi 3NF, tetapi tidak seaiknya. Suatu relasi yang memenuhi 3 NF belum tentu memenuhi BCNF. Dalam banyak literature disebutkan bentuk normal ketiga pun mungkin masih mengandung anomali sehingga masih perlu dinormalisasikan selanjutnya (Janner Simarmata; 2007: 75-83).

II.4. *Entity Relationship Diagram* (ERD)

Suatu model data adalah suatu penyajian konseptual dari suatu data yang diperlukan oleh basis data. Struktur data meliputi objek data, asosiasi antarobjek data, dan aturan yang memerintah operasi pada objek. Seperti yanmg tersirat pada namanya, model data berfokus pada data apa yang diperlukan dan bagaimana data

tersebut harus diorganisasikan, alih-alih pada apa yang dilakukan pada data tersebut. Sebagai analogi, model data setara dengan gambar perencanaan yang dibuat oleh seorang arsitek.

Suatu model data todak terkait pada perangkat keras atau perangkat lunak. Dari pada mencoba menyajikan data sebagai basis data yang akan terlihat, model data berfokus untuk mewakili data yang dilihat pengguna di " dunia nyata". Model data bertindak sebagai jembatan antara konsep yang menyusun dunia nyata dan proses serta tampilan fisik dari konsep tersebut di dalam suatu basis data (Janner Simarmata; 2007: 94).

II.4.1. Langkah- Langkah Perancangan ER

1. Memilih kelompok atribut yang sama untuk dijadikan sebuah entitas dan menentukan kunci utama dengan syarat unik serta bisa mewakili entitas.
2. Menggambarkan kardinalitas (*cardinality*) dan diagram ER berdasarkan relasi yang diperlukan. Relasi yang terjadi adalah hubungan satu- kesatu, satu- kebanyak, dan banyak- kebanyak.
3. Membentuk Skema database atau LRS (*Logical Record Structure*) berdasarkan diagram ER.
    a. Jika relasinya satu- kesatu, maka *foreign key* diletakkan pada salah satu dari 2 entitas yang ada atau menyatukan kedua entitas tersebut.
    b. Jika relasinya satu- kebanyak, maka *foreign key* diletakkan pada entitas Many.
    c. Jika relasinya banyak- kebanyak, maka dibuat "file konektor" pada berisi 2 *foreign key* yang berasal dari kedua entitas.

4. Membentuk beberapa tabel berdasarkan primary key yang terpilih dengan syarat sudah mencapai aturan normalisasi sekurang-kurangnya 3 NF dari skema DB/LRS yang ada (Janner Simarmata; 2007: 115).

II.5. *Unified Modeling Language* (UML)

UML singkatan dari *Unified Modelling Langguage* yang berarti bahasa pemodelan standart. (Chonoles; 2003 : 6) mengatakan sebagai bahasa, berarti *UML* memiliki sintaks dan *semantic*. Ketika kita membuat model menggunakan konsep *UML* ada aturan–aturan yang harus diikuti. Bagaimana elemen pada model-model yang kita buat harus berhubungan satu dengan lainnya harus mengikuti standart yang ada. *UML* bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. Ketika pelanggan memesan sesuatu dari sistem, bagaimana transaksinya? Bagaimana sistem mengatasi error yang terjadi? Bagaimana keamanan terhadap sistem yang ada kita buat? Dan sebagainya dapat dijawab dengan *UML*.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk :

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.

UML telah diaplikasikan dalam investasi perbankan,lembaga kesehatan, departemen pertahanan, sistem terdistribusi, sistem pendukung alat kerja, retail, sales, dan supplier.

Blok pembangunan utama *UML* adalah diagram. Beberapa diagram ada yang rinci (jenis *timing diagram*) dan lainya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorentasikan objek menggunakan bahasa model untuk menggambarkan, membangun dan mendokumentasikan sistem yang mereka rancang. *UML* memungkinkan para anggota team untuk bekerja sama dalam mengaplikasikan beragam sistem. Intinya, *UML* merupakan alat komunikasi yang konsisten dalam mensuport para pengembang sistem saat ini. Sebagai perancang sistem mau tidak mau pasti menjumpai *UML*, baik kita sendiri yang membuat sekedar membaca diagram *UML* buatan orang lain (Prabowo Pudji Widodo, Herlawati; 2011 : 6-7).

II.5.1. Diagram-*Diagram UML*

Beberapa literatur menyebutkan bahwa *UML* menyediakan Sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa yang digabung, misalnya diagram komunikasi, diagram urutan, dan diagram pewaktuan digabung menjadi diagram interaksi. Namun demikian model-model itu dapat dikelompokan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain :

1. Diagram Kelas. Bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi, serta relasi-relasi diagram. Diagram ini umu dijumpai pada pemodelan sistem berorentasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas.

2. Diagram Paket (*Package Diagram*) bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas merupakan bagian dari diagram komponen.
3. Diagram *Use Case* bersifat statis. Diagram ini memperlihatkan himpunan *use-case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna.
4. Diagram Interaksi dan *Sequence* (urutan). Bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam waktu tertentu.
5. Diagram Komunikasi (*Communication Diagram*) bersifat dinamis. Diagram sebagai pengganti diagram kolaborasi *UML* yang menekankan organisasi *structural* dari objek-objek yang menerima serta mengirim pesan.
6. Diagram *Statechart* (*Statechart Diagram)* bersifat dinamis. Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status (*State),* transisi kejadian serta aktifitas. Diagram ini terutama penting untuk memperlihatkan sifat dinamis dari antarmuka (*interface*), kelas, kolaborasi dan terutam penting pada pemodelan sistem-sistem yang reaktif.
7. Diagram Aktivitas (*Activity Diagram*) bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi suatu sistem dan member tekanan pada aliran kendali antar objek.

8. Diagram Komponen (*Component Diagram*) bersifat statis. Diagram komponen ini memperlihatkan organisasi serta kebergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan diagram kelas dimana komponen dipetakan kedalam satu atau lebih kelas-kelas. Antarmuka-antarmuka serta kolaborasi-kolaborasi.
9. Diagram *Deployment* (*Deployment Diagram)* bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*run time)*. Memuat simpul-simpul berserta komponen-komponen yang ada di dalamnya. Diagram *Deployment* berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing).*

   Kesembilan diagram ini tidak mutlak harus digunakan dalam pengembangan perangkat lunak, semuanya dibuat sesuai dengan kebutuhan. Pada *UML* dimungkinkan kita menggunakan diagram-diagram lainya misalnya *Data Flow Diagram, Entity Relationship Diagram* dan sebagainya (Probowo Pudji Widodo, Herlawati; 2011 : 10-12).

II.5.2. Diagram Use Case (Use Case Diagram)

1. *Diagram Use Case (Use Case Diagram)*

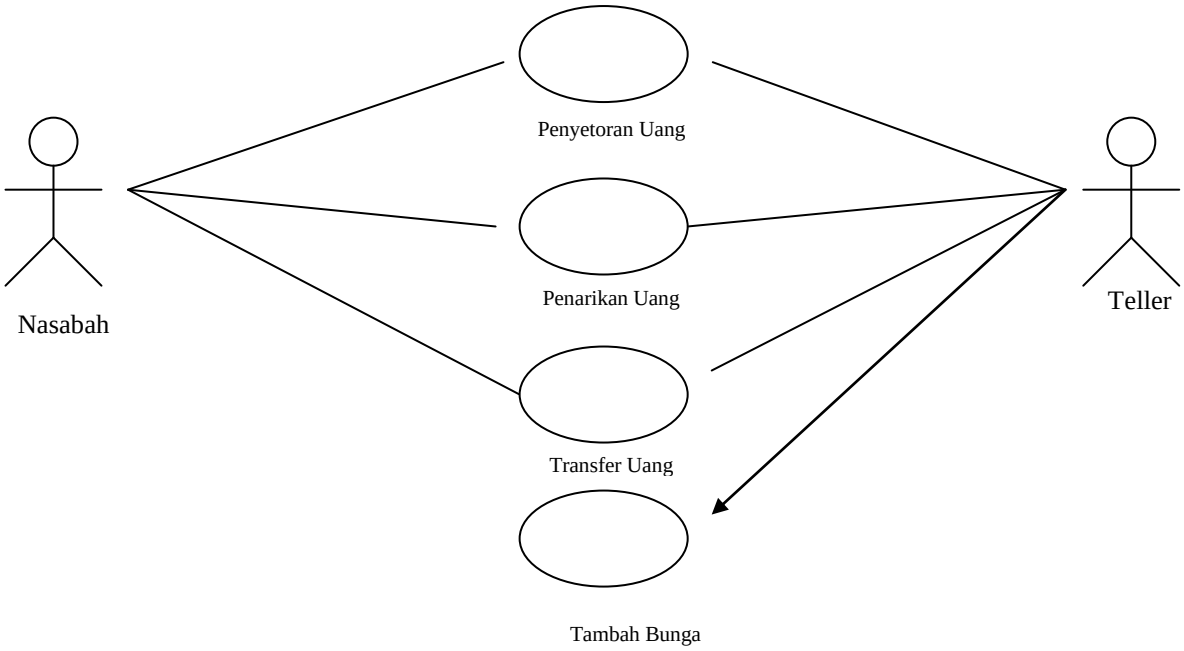
   Use Case menggambarkan *external view* dari sistem yang akan kita buat modelnya. Menurut Pooley (2005:15) mengatakan bahwa model *use case*

dapat dijabarkan dalam diagram, tetapi yang perlu diingat, diagram tidak indentik dengan model karena model lebih luas dari diagram.

komponen pembentuk diagram *use case* adalah :

a. Aktor (*actor)*, menggambarkan pihak-pihak yang berperan dalam sistem.
b. *Use Case*, aktivitas/ sarana yang disiapkan oleh bisnis/sistem.
c. Hubungan (*Link)*, aktor mana saja yang terlibat dalam *use case* ini.

Gambar II.1. di bawah ini merupakan salah satu contoh bentuk diagram *use case* (Prabowo Pudji Widodo, Herlawati; 2011 : 15-16).



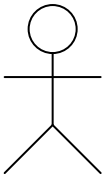
Gambar II.1. Diagram *Use Case*

(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:17)

2. Aktor

Menurut Chonoles (2003 :17) menyarankan sebelum mebuat use case dan menentukan aktornya, agar mengidentifikasi siapa saja pihak yang terlibat

dalam sistem kita. Pihak yang terlibat biasanya dinamakan *stakeholder* untuk melihat gambar aktor, lihat pada gambar II.2. sebagai berikut :

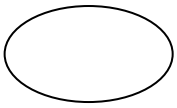


Gambar II.2. Aktor

(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:17)

3. *Use Case*

Menurut Pilone (2005 : 21) *use case* menggambarkan fungsi tertentu dalam suatu sistem berupa komponen kejadian atau kelas. Sedangkan menurut Whitten (2004 : 258) mengartikan *use case* sebagai urutan langkah-langkah yang secara tindakan saling terkait (skenario) baik terotomatisasi maupun secara manual, untuk tujuan melengkapi satu tugas bisnis tunggal. *Use case* digambarkan dalam bentuk *ellips*/oval untuk melihat gambar simbol *use case*, lihat pada gambar II.3. sebagai berikut :



Gambar II.3. Simbol *Use Case*

(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:22)

Use case sangat menentukan karakteristik sistem yang kita buat, oleh karena itu Chonoles (2003:22-23) menawarkan cara untuk menghasilkan *use case* yang baik yakni :

a. Pilihlah Nama Yang Baik

Use case adalah sebuah *behaviour* (prilaku), jadi seharusnya dalam frase kata kerja. Untuk membuat namanya lebih detil tambahkan kata benda mengindikasikan dampak aksinya terhadap suatu kelas objek. Oleh karena itu diagram *use case* seharusnya berhubungan dengan diagram kelas.

b. Ilustrasikan Perilaku Dengan Lengkap.

Use case dimulai dari inisiasi oleh aktor primer dan berakhir pada aktor dan menghasilkan tujuan. Jangan membuat *use case* kecuali anda mengetahui tujuannya. Sebagai contoh memilih tempat tidur (*King Size, Queen Size,* atau dobel) saat tamu memesan tidak dapat dijadikan *use case* karena merupakan bagian dari *use case* pemesanan kamar dan tidak dapat berdiri sendiri (tidak mungkin tamu memesan kamar tidur jenis king tapi tidak memesan kamar hotel).

c. Identifikasi Perilaku Dengan Lengkap.

Untuk mencapai tujuan dan menghasilkan nilai tertentu dari aktor, use case harus lengkap. Ketika memberi nama pada *use case*, pilihlah frasa kata kerja yang implikasinya hingga selesai. Misalnya gunakan frasa *reserve a room* (pemesanan kamar) dan jangan *reserving a room* (memesan kamar) karena memesan menggambarkan perilaku yang belum selesai.

d. Menyediakan *Use Case* Lawan (*Inverse)*

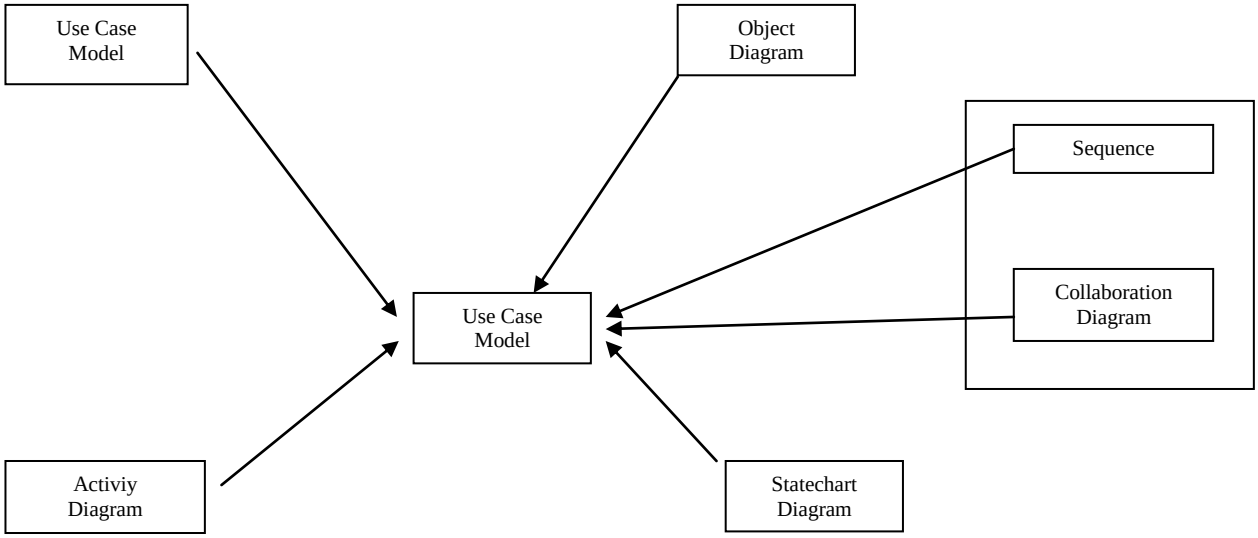
Kita biasanya membutukan *use case* yang membatalkan tujuan, misalnya pada us*e case* pemesanan kamar, dibutuhkan pula *use case* pembatalan pesanan kamar.

e. Batasi Use Case Hingga Satu Perilaku Saja.

Kadang kita cenderung membuat *use case* yang lebih dari satu tujuan aktivitas. Guna menghindari kerancuan, jagalah use case kita hanya fokus pada satu hal. Misalnya, penggunaan *use case check in* dan *check out* dalam satu *use case* menghasilkan ketidakfokusan, karena memiliki dua perilaku yang berbeda (Prabowo Pudji Widodo, Herlawati; 2011 : 22-23)

4. Diagram Kelas (*Class Diagram*)

Diagram kelas adalah inti dari proses pemodelan objek. Baik *forward engineering* maupun *reverse engineering* memanfaatkan diagram ini *forward engineering* adalah proses perubahan model menjadi kode program sedangkan *reverse engineering* sebaliknya merubah kode program menjadi model (Probowo Pudji Widodo, Herlawati; 2011 : 37) untuk melihat gambar hubungan diagram kelas dengan diagram *uml* lainya, lihat pada gambar II.4. sebagai berikut :



Gambar II.4. Hubungan Diagram Kelas Dengan Diagram *UML* lainya
(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011 : 38)

5. Diagram Aktivitas (*Activity Diagram*)

Diagram aktivitas lebih memfokuskan diri pada eksekusi dan alur sistemdari pada bagaimana sistem dirakit. Diagram ini tidak hanya memodelkan software melainkan memodelkan bisnis juga. Diagram aktivitas menunjukan aktivitas sistem dalam kumpulan aksi-aksi. Ketika digunakan dalam pemodelan *software*, diagram aktivitas merepresentasikan pemanggilan suatu fungsi tertentu misalnya *call*. Sedangkan bila digunakan dalam pemodelan bisnis, diagram ini menggambarkan aktivitas yang dipicu oleh kejadian-kejadian diluar seperti pemesanan atau kejadian-kejadian internal misalnya penggajian tiap jumat sore (Prabowo Pudjo Widodo, Herlawati ;2011 : 143-145)

Aktivitas merupakan kumpulan aksi-aksi. Aksi-aksi nelakukan langka sekali saja tidak boleh dipecah menjadi beberapa langkah-langkah lagi. Contoh aksinya yaitu :

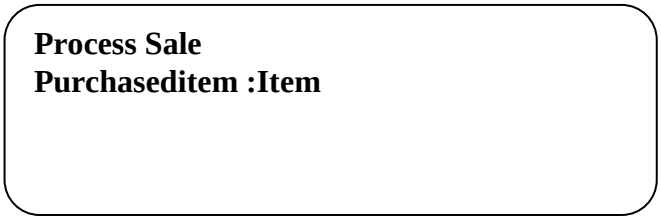
a. Fungsi Matematika

b. Pemanggilan Perilaku

c. Pemrosesan Data

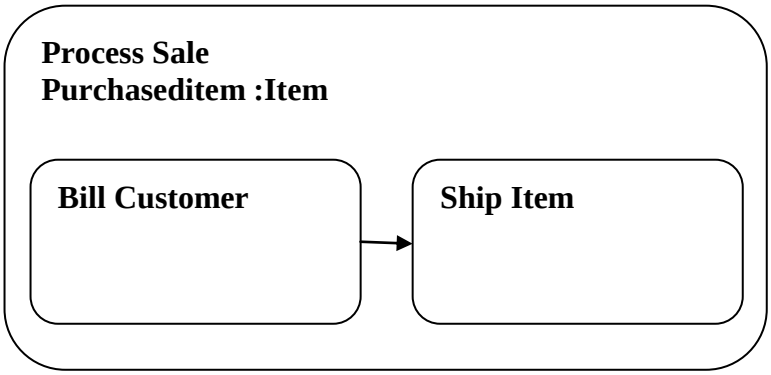
Ketika kita menggunakan diagram aktivitas untuk memodelkan perilaku suatu *classfier* dikatakan kontek dari aktivitas. Aktivitas dapat mengakses atribut dan operasi *classfier*, tiap objek yang terhubung dan parameter-parameter jika aktivitas memiliki hubungan dengan perilaku. Ketika digunakan dengan model proses bisnis, informasi itu biasanya disebut *process-relevant data*. Aktivitas diharapkan dapat digunakan ulang dalam suatu aplikasi, sedangkan aksi biasanya *specific* dan digunakan hanya untuk aktivitas tertentu.

Aktivitas digambarkan dengan persegi panjang tumpul. Namanya ditulis di kiri atas. Parameter yang terlibat dalam aktivitas ditulis dibawahnya. Untuk melihat gambar aktivitas serderhana tanpa rincian, lihat pada gambar II.5. sebagai berikut :



Gambar II.5. Aktivitas Serderhana Tanpa Rincian
(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:145)

Detail aktivitas dapat dimasukan di dalam kotak. Aksi diperlihatkan dengan symbol yang sama dengan aktivitas dan namanya diletakkan didalam persegi panjang. Untuk melihat gambar aktivitas dengan detail rincian, lihat pada gambar II.6. sebagai berikut :



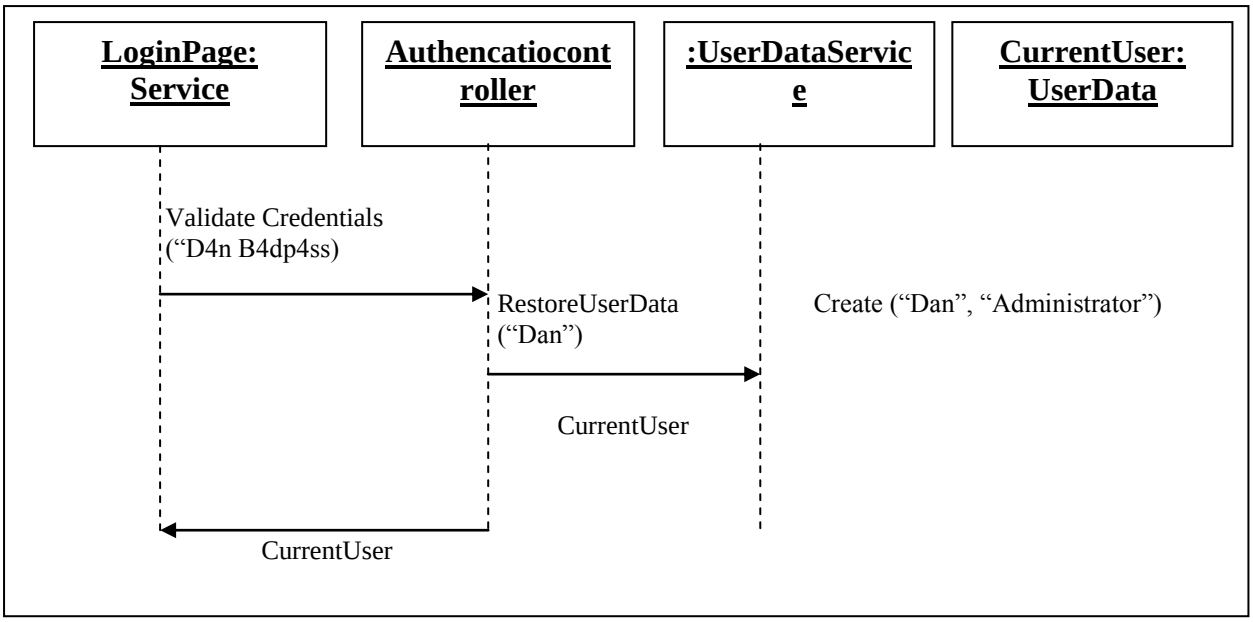
Gambar II.6. Aktivitas Dengan Detail Rincian

(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:145)

6. *Sequence Diagram*

Menurut Douglas (2004 : 174) menyebutkan ada tiga diagram primer UML dalam memodelkan scenario interaksi, yaitu diagram urutan (*sequence diagram)*, diagram waktu (*timing diagram)* dan diagram komunikasi (*communication diagram)*.

Menurut Pilone (2005 : 174) menyatakan bahwa diagram yang paling banyak dipakai adalah diagram urutan. Gambar II.7. memperlihatkan contoh diagram urutan dengan notasi-notasinya yang akan dijelaskan nantinya (Prabowo Pudji Widodo, Herlawati; 2011 : 174-175)



Gambar II.7. Diagram Urutan

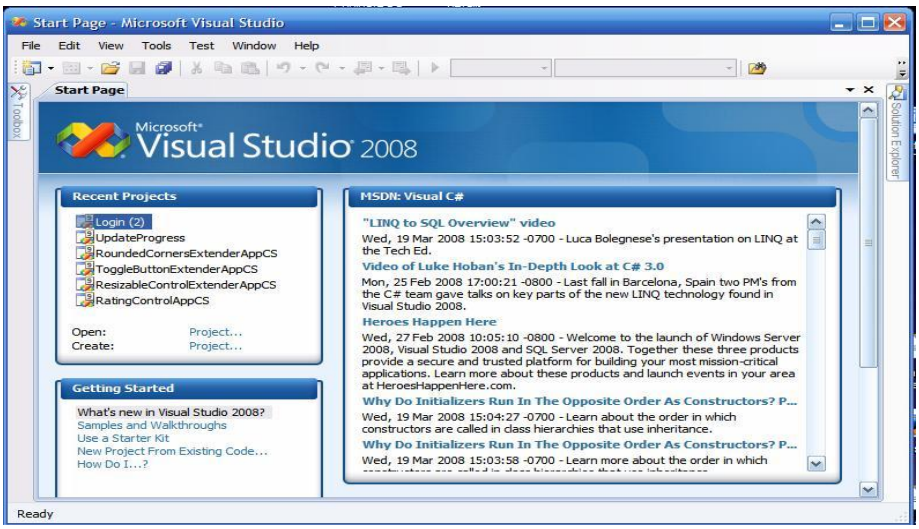
(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:175)

II.6. Bahasa Pemograman Microsoft Visual Studio 2008

Microsoft Visual Studio 2008 merupakan kelanjutan dari *Microsoft Visual Studio* sebelumnya, yaitu *Visual Studio. Net* 2003 yang diproduksi oleh Microsoft. Pada bulan Februari 2002 *Microsoft* memproduksi teknologi. *Net Framework* versi 1.0, teknologi. *Net* ini didasarkan atas susunan berupa *Net Framework*, sehingga setiap produk baru yang terkait dengan teknologi. Net akan selalu berkembang mengikuti perkembangan. *Net Frameworknya*. Pada perkembangannya nantinya mungkin untuk membuat program dengan teknologi. *Net* memungkinkan para pengembang perangkat lunak akan dapat menggunakan lintas sistem operasi, yaitu dapat dikembangkan di sistem operasi windows juga dapat dijalankan pada sistem operasi lain, misalkan pada sistem operasi *Linux*, seperti yang telah dilakukan pada pemograman *Java* oleh *Sun Microsystem*. Pada saat ini perusahaan-perusahaan sudah banyak meng*update* aplikasi lama yang

dibuat *Microsoft Visual Basic* 6.0 ke teknologi. *Net* karena kelebihan-kelebihan yang ditawarkan, terutama memungkinkan pengembang perngkat lunak secara cepat mampu membuat program *robust*, serta berbasiskan integrasi ke internet yang dikenal dengan *XML Web Service* (Wahana Komputer ; 2008 : 1)

Untuk melihat tampilan visual studio 2008 dapat dilihat pada gambar II.8. sebagai berikut :



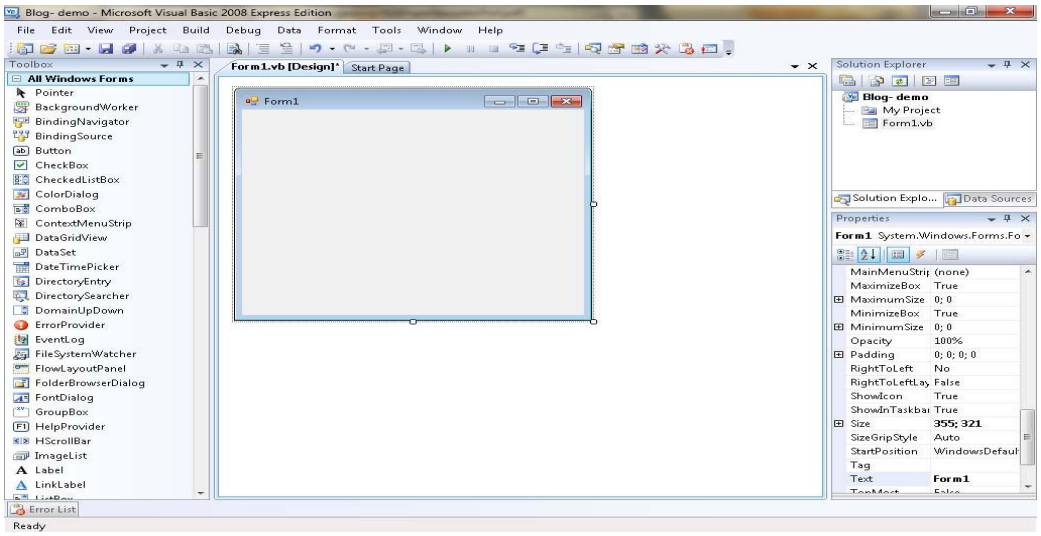
Gambar II.8. Tampilan Utama Visual Studio 2008

(Sumber : Wahana Komputer; 2008 : 12)

II.6.1. Antar Muka *Microsoft Visual Basic. Net IDE* 2008.

Antarmuka atau lingkungan dari *Visual Basic. Net IDE* 2008 tidak jauh berbeda dengan *Visual Basic 6.0 IDE*, kelebihannya memiliki IDE (*Interface Development Envroitment*) yang lebih lengkap dan terorganisasi, sehingga mudah bagi pengembang untuk mencari objek-objek atau komponen yang terdapat pada *toolbox* yang kita inginkan, untuk ditempatkan pada objek *form*, dengan *meng-klik sebuah objek* dan kemudian diletakkan diatas *form*. Berikut adalah tampilan

lingkungan dari *Visual Basic. Net* 2008 dapat dilihat pada Gambar II.9. sebagai berikut :



Gambar II.9. Interface Microsoft Visual Basic. Net IDE 2008

(Sumber : Wahana Komputer; 2008 : 13)

II.6.2. Lingkungan Kerja Pada Microsoft Visual Basic. Net 2008

1. *Tittle Bar*

   Tittle bar berfungsi untuk menampilkan nama project yang aktif atau sedang dikembangkan. Adapun *tittle bar* dapat dilihat pada gambar II.10. sebagai berikut :

Gambar II.10. Tampilan *Tittle Bar Visual Studio* 2008

(Sumber : Wahana Komputer; 2008 : 14)

2. Menu Bar

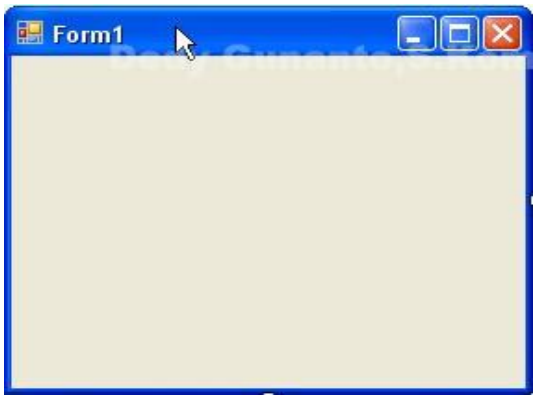
   Menu bar berfungsi untuk pengelolaan fasilitas yang dimiliki oleh *visual basic. Net* 2008, sedangkan *Tool Bar* berfungsi untuk melakukan perintah

khusus secara cepat. Adapun menu bar dapat dilihat pada gambar II.11. sebagai berikut :

Gambar II.11. Tampilan Menu Bar Visual Studio 2008
(Sumber : Wahana Komputer; 2008 : 14)

3. *Form*

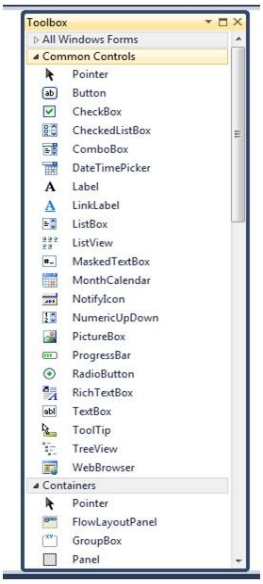
Form adalah objek utama yang berfungsi untuk meletakkan objek-objek yang terdapat pada toolbox yang digunakan dalam melakukan perancangan sebuah tampilan program aplikasi. Adapun *form* dapat dilihat pada gambar II.12. sebagai berikut :



Gambar II.12. Tampilan *Form Visual Studio* 2008
(Sumber : Wahana Komputer; 2008 : 14)

4. *ToolBox*

ToolBox berfungsi untuk menyediakan objek-objek atau komponen yang digunakan dalam merancang sebuah *form* pada program aplikasi. Adapun tampilan *toolbox* dapat dilihat pada gambar II.13. sebagai berikut :

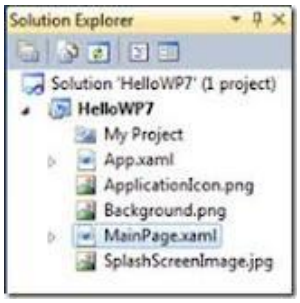


Gambar II.13. Tampilan *ToolBox Visual Studio* 2008

(Sumber : Wahana Komputer; 2008 : 15)

5. *Solution Explorer*

Solution Explorer berfungsi untuk menampilkan *project* beserta *file-file* pendukung yang terdapat pada sebuah program aplikasi. Adapun tampilan *Solution Explorer* dapat dilihat pada gambar II.14. sebagai berikut :

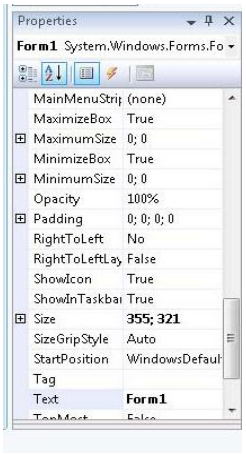


Gambar II.14. Tampilan *Solution Explorer Visual Studio* 2008

(Sumber : Wahana Komputer; 2008 : 16)

6. *Properties Windows*

Properties windows berfungsi untuk mengatur *properties-properties* pada objek (*setting object*) yang diletakkan pada sebuah *form*. Adapun tampilan *properties windows* dapat dilihat pada gambar II.15. sebagai berikut :



Gambar II.15. Tampilan *Properties Windows Visual Studio* 2008
(Sumber : Wahana Komputer; 2008 : 16)

II.7. *Microsoft SQL Server*

SQL Server Management Studio membantu anda mengatur database dengan mudah. Anda dapat melakukan pengaturan atas beberapa pada sebuah komputer saja atau melakukan pengaturan *server* secara *remote*. Anda dapat juga membuat *database*, *table*, *index*, dan melakukan manipulasi data terhadap *database* dan tabel-tabelnya.

SQL Server Management Studio memiliki beberapa komponen penting yang mewakili kegunaanya dalam perancangan *database*, dan melakukan pengaturan sistem secara keseluruhan. Komponen-komponen tersebut itu adalah :

a. *Registered Server*

b. *Object Explorer*

c. *Query editor*

Adapun tampilan Microsoft SQL Server 2008 dapat dilihat pada gambar.16. sebagai berikut (Wahana Komputer; 2008 : 40)

Gambar II.15. *SQL Server 2008*

Sumber : Wahana Komputer (2008 : 40)

II.7.1. *Interface SQL Server 2008*

Ada 3 *interface* utama saat bekerja dengan *SQL Server* 2008 adalah sebagai berikut :

1. *Registered Server*

Bila pada tampilan pertama anda tidak melihat panel ini maka anda dapat menampilkannya pada menu *View Regristed Servers* atau menekan kombinasi tombol CTRL + ALT + G. Panel ini memungkinkan anda menjaga koneksi-koneksi dengan server-server yang pernah digunakan. Koneksi-koneksi ini dapat digunakan untuk memeriksa dari server tersebut (*online* atau *offline*) atau melakukan pada obyek-obyeknya (menggunakan pada panel *object explorer*). Setiap user memiliki daftar tersendiri dari

regristered server yang disimpan pada mesin lokal. Anda dapat melakukan penambahan atau pengurangan koneksi ke *server*. Anda dapat mengkelompokan koneksi – koneksi ke server tersebut berdasarkan tipe servernya yaitu *Database Engine, Analysis Service, Reporting Service, Intergration Service.*

2. *Object Explorer*

   Anda dapat melihat berbagai obyek yang ada pada sebuah server pada panel ini. Bila panel ini tidak terlihat maka anda dapat menampilkannya dengan menu *View Object Explorer*. Apabila anda melakukan ekpansi dari sebuah cabang maka sebuah struktur logika dari sebuah obyek akan muncul. Anda dapat mengklik tanda + pada sebelah kiri untuk melakukan ekspansi cabang. Untuk melakukan koneksi pada sebuah server klik kanan pada nama servernya kemudian pilih *Connect*, untuk melakukan *Disconnect* , klik kanan dan pilih *disconnect*.

3. *Query Editor*

   Jendela ini digunakan untuk membuat, melakukan editing perintah perintah T-SQL dan mengeksekusi perintah tersebut. Jendela ini akan muncul otomatis setiap kali anda melakukan kegiatan yang berhubungan dengan query. Apabila anda berminat membuat *query* baru atau melakukan *editing file query* yang telah ada, maka jendela ini otomatis akan muncul. Anda dapat membuat *File Query With Current Connection* atau *Database Engine Query atau SQL Server Commpact Query* atau memanfaatkan tombol *New Query* pada *toolbar* (Wahana Komputer ; 2008 : 45-49).

II.7.2. Komponen –Komponen *SQL Server* 2008

Ada 5 komponen-komponen SQL Server 2008 adalah sebagai berikut :

1. *Literal Value*

   Yang termasuk dengan *literal value* adalah huruf (a – z), *numerik* (0-9), dan hexadesimal (0x). *Literal value* juga dikenal dengan konstanta. Sebuah *konstanta string* terdiri atas satu atau beberapa karakter yang diapit tanda kutip tunggal *(apostroph)* atau tanda petik ganda. Defenisi *konstanta string* sebaiknya digunakan dengan tanda petik tunggal karena tanda petik ganda dalam *query* memiliki fungsi lain.

2. Delimeter

   Bahwa sebaiknya menggunakan tanda petik tunggal untuk mengawali dan mengakhiri sebuah konstanta *string* dari pada tanda petik ganda. Hal ini karena tanda petik ganda juga digunakan sebagai delimeter atau pemisah. Tanda kutip ganda tidak dapat digunakan sebagai pembuka dan penutup dari sebuah konstanta string perintah berikut ini diberikan yaitu *Set Queted Identifier On* Standar perinth tersebut adalah *ON*. Perintah tersebut mengakibatkan tanda petik ganda diatur menjadi *Delimeted Identifier*. *Delimeted Identifier* adalah *identifier* khusus yang memungkinkan reserver word digunakan menjadi *identifier* dan juga membolehkan adanya spasi pada nama *database.*

3. Komentar

   Komentar dalam pemograman ataupun *scripting* diperlukan untuk memberikan keterangan singkat tentang kode-kode yang ada dibawahnya.

Sehingga sewaktu ada kerusakan, kesalahan, *programmer* dapat dengan mudah menngerti apa kegunaan dari kode tersebut pada *SQL Server* terdapat dua macam komentar yaitu :

a. Komentar yang menggunakan tanda / * * /. Dengan tanda ini komentar yang anda berikan dapat terdiri atas beberapa baris di awali dengan / * dan di akhiri dengan */.

b. Komentar yang menggunakan tanda – untuk memberi komentar pada baris yang dimaskud saja. Biasanya digunakan untuk menerangkan *identifier* atau *reserved word*.

4. *Identifier*

Dalam pemograman bahasa *T-SQL* untuk *query*, *identifier* digunakan untuk melakukan identifikasi *database* dan obyek-obyeknya seperti tabel dan *index*. Diidentifikasi dengan string karakter dengan panjang maksimal 128 karakter. *Identifier* ini dapat terdiri atas berbagai macam huruf, angka dan karakter khusus (@, _#, $). Setiap identifier harus dimulai dengan huruf, atau karakter khusus tidak boleh dengan angka.

5. *Reserved Word*.

Reserved word atau kata kunci adalah kata yang memilikiarti khusus dan harus dituliskan dengan aturan tertentu. Dalam bahasa *T-SQL reserved word* dan juga memiliki banyak fungsi. Reserved word tidak dapat digunakan sebagai nama sebuah obyek kecuali obyek tersebut didefenisikan sebagai *delimeted identifier*. (Wahana Komputer ; 2008: 84-86).