

BAB II

TINJAUAN PUSTAKA

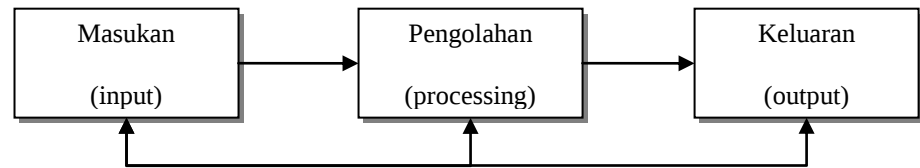
II.1. Konsep dasar Sistem

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

II.1.1. Sistem

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. (Hanif Al Fattah, 2010 : 3).

Menurut *Scott (1996)*, sistem terdiri dari unsur-unsur seperti masukan (*input*), pengolahan (*processing*), serta keluaran (*output*). Ciri pokok sistem menurut Gaspert ada empat, yaitu sistem itu beroperasi dalam suatu lingkungan, terdiri atas unsur-unsur, ditandai dengan saling berhubungan, dan mempunyai satu fungsi atau tujuan utama.

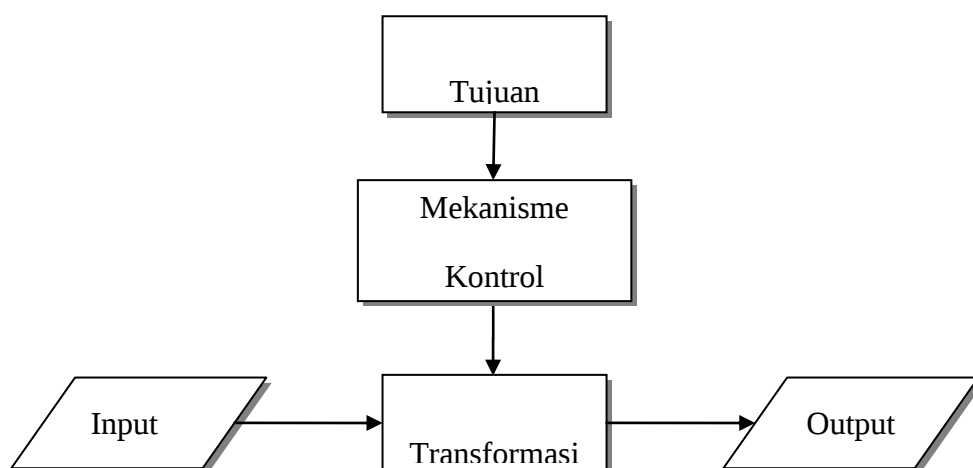


Gambar II.1. Model Sistem

Sumber : Hanif Al Fattah (2010 : 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 Fattah (2010 : 4)

Banyak ahli mengajukan konsep sistem dengan deskripsi yang berbeda, tetapi pada prinsipnya hampir sama dengan konsep dasar sistem umumnya. *Schröderberg (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, 2010 : 5).

II.1.2. Karakteristik Sistem

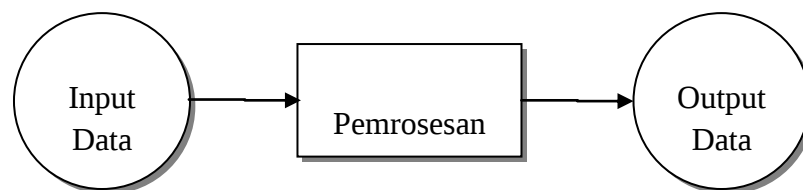
Untuk memahami atau mengembangkan suatu sistem, maka perlu membedakan unsur-unsur dari sistem yang membentuknya. Berikut adalah karakteristik sistem yang dapat membedakan suatu sistem dengan sistem yang lainnya :

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 layar komputer, 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, 2010 : 5 - 6).

II.1.3. Sistem Informasi

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*). *McLeod (1995)* mengatakan bahwa informasi adalah data yang telah diproses, atau data yang memiliki arti (*Hanif Al Fattah, 2010 : 9*).

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 berikut :



Gambar II.3. Konsep sistem informasi

Sumber : Hanif Al Fattah (2010 : 9)

II.2. Akuntansi

Terdapat beberapa definisi akuntansi , diantaranya adalah :

1. Menurut *Accounting Principles Board* (1970) :

Akuntansi adalah suatu kegiatan jasa yang fungsinya menyediakan informasi kuantitatif, terutama yang bersifat keuangan tentang entitas ekonomi yang dimaksudkan agar berguna dalam mengambil keputusan ekonomi – membuat pilihan-pilihan nalar diantara berbagai alternatif arah tindakan.

2. Menurut *American Accounting Association* (1996) :

Akuntansi adalah suatu proses pengidentifikasian, pengukuran, pencatatan, dan pelaporan transaksi ekonomi (keuangan) dari suatu organisasi/ entitas yang dijadikan sebagai informasi dalam rangka mengambil keputusan ekonomi oleh pihak-pihak yang memerlukan.

Dalam kedua defini tersebut diketahui bahwa :

1. Fungsi (peran) akuntansi adalah menyediakan informasi kuantitatif, terutama yang bersifat keuangan, tentang “entitas” ekonomi.
2. Informasi yang dihasilkan oleh akuntansi dimaksudkan agar berguna sebagai input yang dipertimbangkan dalam mengambil keputusan ekonomi yang rasional (Abdul Halim : 2010 : 32).

II.3. Sistem Informasi Akuntansi

Sistem informasi akuntansi merupakan sekumpulan sumber daya seperti manusia dan peralatan, yang diatur untuk mengubah data menjadi informasi. Informasi ini dikomunikasikan kepada beragam pengambil keputusan.

Sistem informasi akuntansi pada umumnya meliputi beberapa siklus pemrosesan transaksi :

1. Siklus pendapatan. Berkaitan dengan pendistribusian barang dan jasa ke entitas lain dan pengumpulan pembayaran-pembayaran yang berkaitan.
2. Siklus pengeluaran. Berkaitan dengan perolehan barang jasa dari entitas lain dan pelunasan kewajiban yang berkaitan.
3. Siklus produksi. Berkaitan dengan perubahan sumberdaya menjadi barang dan jasa.
4. Siklus keuangan. Kejadian-kejadian yang berkaitan dengan perolehan dan manajemen dana-dana modal, termasuk kas (Mujilan : 2012 : 3).

Fungsi penting yang dibentuk SIA pada sebuah organisasi antara lain :

1. Mengumpulkan dan menyimpan data tentang aktivitas dan transaksi.
2. Memproses data menjadi informasi yang dapat digunakan dalam proses pengambilan keputusan.

3. Melakukan kontrol secara tepat terhadap aset organisasi (Tata Sutabri : 2012 : 83).

Subsistem SIA memproses berbagai transaksi keuangan dan transaksi non keuangan yang secara langsung mempengaruhi pemrosesan transaksi keuangan. Sistem Informasi Akuntansi terdiri dari 2 subsistem yaitu :

1. Sistem pemrosesan transaksi yang mendukung proses operasi bisnis harian.
2. Sistem buku besar / pelaporan keuangan yang menghasilkan laporan keuangan, seperti laporan laba / rugi, neraca, arus kas, dan pengembalian pajak (Tata Sutabri : 2012 : 83).

Tujuan sistem informasi akuntansi adalah untuk menyediakan informasi yang diperlukan dalam pengambilan keputusan yang dilaksanakan oleh aktivitas yang disebut pemrosesan informasi. Sebagian dari keluaran yang diperlukan oleh pemroses informasi disediakan oleh sistem pemrosesan transaksi, seperti laporan keuangan dari sistem pemrosesan transaksi (Tata Sutabri : 2012 : 84).

II.4. Penjualan

Penjualan merupakan tujuan utama dilakukannya kegiatan perusahaan. Perusahaan dalam menghasilkan barang dan jasa mempunyai tujuan akhir, yaitu menjual barang dan jasa tersebut kepada masyarakat. Oleh karena itu, penjualan memegang peranan penting bagi perusahaan agar produk yang dihasilkan oleh perusahaan dapat terjual dan memberikan penghasilan bagi perusahaan. Penjualan

yang dilakukan perusahaan bertujuan untuk menjual barang atau jasa yang diperlukan sebagai sumber pendapatan untuk menutup semua ongkos untuk memperoleh laba.

Penjualan adalah pemindahan hak milik atas barang atau pemberian jasa yang dilakukan penjualan kepada pembeli dengan harga yang disepakati bersama dengan jumlah yang dibebankan kepada pelanggan dalam penjualan barang dan jasa dalam suatu periode akuntansi.

Penjualan merupakan pengalihan hak milik atas barang dengan imbalan uang sebagai gantinya dengan persetujuan untuk menyerahkan barang kepada pihak lain dengan menerima pembayaran (Freddy Rangkuti : 2010 : 57).

Sistem informasi akuntansi penjualan dapat menghasilkan jurnal umum, laporan data-data penjualan, dan peramalan rata-rata penjualan barang dagang.

II.5. Metode Average

Rata-rata (*average*), pengeluaran barang secara acak dan harga pokok barang yang sudah digunakan maupun yang masih ada ditentukan dan dicari rata-ratanya (Arseto Pramono dan Moch Arifin : 2010 : 3).

Untuk menghitung persediaan akhir dan harga pokok penjualan perlu dibuat perhitungan sebagai berikut :

Tabel II.1. Perhitungan Persediaan Akhir

Tanggal	Keterangan	Unit	Harga Per Unit	Jumlah
Jan 1	Persediaan	200	\$ 10	\$2.000
12	Pembelian	400	\$12	\$4.800
26	Pembelian	300	\$11	\$3.300
30	Pembelian	100	\$13	\$1.300
Jumlah		1.000		\$11.400
Rata-Rata = \$11.400 : 1000				\$11.4

Harga pokok penjualan = $700 \times \$11.4 = \7.980

Persediaan akhir = $300 \times \$11.4 = 3.240$

(Jurnal Teknologi Informasi – Aiti ; Arseto Pramono, dkk ; 2010 : 3).

Metode ini sering disebut dengan istilah metode rata-rata tertimbang (*weight average method*), didasarkan atas anggapan bahwa *cost* yang harus diperhitungkan untuk menentukan *cost* sediaan sediaan akhir dan *cost* barang yang terjual dengan menggunakan *cost* rata-rata per unit.

Tabel II.2. Perhitungan HPP Persediaan Menggunakan Metode Average

Tanggal	Masuk			Keluar			Saldo		
	Qty	Hpp	Nilai	Qty	Hpp	Nilai	Qty	Hpp	Nilai
1 Jan Sediaan	200	105	21.000				200	105	21.000
5 Jan Pembelian	350	110	38.500				550	107,5	21.000
8 Jan Penjualan 150 Unit				150	107,5	16.125	400	107,5	43.000
25 Jan Pembelian	200	130	26.000				600	118,75	71.250
28 Jan Penjualan 500 Unit				500	118,75	59.375	100	118,75	11.875

Hpp penjualan untuk bulan januari = $16.125 + 59.375 = 75.500$

II.6. Entity Relationship Diagram (ERD)

ERD (*Entity Relationship Diagram*) adalah gambar atau diagram yang menunjukkan informasi yang dibuat, disimpan, dan digunakan dalam sistem bisnis. Entitas biasanya menggambarkan jenis informasi yang sama. Dalam entitas digunakan untuk menghubungkan antar entitas yang sekaligus menunjukkan hubungan antar data. Pada akhirnya ERD juga bisa digunakan untuk menunjukkan aturan-aturan bisnis yang ada pada sistem informasi yang akan dibangun (Hanif Al Fattah ; 2010 ; 121-122).

II.7. Kamus Data

Kamus data (*data dictionary*) mencakup definisi-definisi dari data yang disimpan dalam basis data dan dikendalikan oleh sistem manajemen basis data. Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari seluruh definisi field, definisi tabel, relasi tabel dan hal-hal lainnya. Nama field data, jenis data (seperti teks dan angka atau tanggal), nilai-nilai yang valid untuk data dan karakteristik-karakteristik lainnya akan disimpan dalam kamus data. Perubahan-perubahan pada struktur data hanya dilakukan satu kali di dalam kamus data; program program aplikasi yang mempergunakan data tidak akan ikut terpengaruh (Raymond McLeod Jr dan George P Schell ; 2010 ; 171).

I.8. Normalisasi

Setelah melalui tahapan di atas atau ERD, maka hasil pada diagram tersebut mulai direlasasikan pada tabel-tabel database. Untuk itu diperlukan sebuah tahapan yang disebut normalisasi. Normalisasi data adalah proses di mana tabel-tabel pada database dites dalam hal kesalingtergantungan di antara field-

field pada sebuah tabel. Misalnya jika pada sebuah tabel terdapat ketergantungan terhadap lebih dari satu field dalam tabel tersebut, maka tabel tersebut harus dipecah menjadi banyak tabel.

Pada prose normalisasi data, aturan yang dijadikan acuan adalah metode ketergantungan fungsional. Teorinya adalah bahwa tiap kolom dalam sebuah tabel selalu memiliki hubungan yang unik dengan sebuah kolom kunci. Misalnya pada sebuah tabel data_siswa ada field nomor induk data field nama siswa serta field tanggal lahir. Maka ketergantungan fungsionalnya dapat dinyatakan sebagai berikut: $nmr_induk \rightarrow nm_siswa$ dan $nmr_induk \rightarrow tgl_lahir$. Artinya nm_siswa memiliki ketergantungan fungsional terhadap nmr_induk . Field nm_siswa isinya juga ditentukan oleh field nmr_induk . Maksud dari semua itu adalah nmr_induk adalah field kunci yang menentukan karena tidak ada nomor induk yang sama pada satu sekolah, jadi field nmr_induk dapat dijadikan patokan untuk mengisi nm_siswa dan field lainnya. (Wahana Komputer, 2010 : 32)

Ada beberapa langkah dalam normalisasi tabel, yaitu:

1. *Decomposition*, dekomposisi adalah proses mengubah bentuk tabel supaya memenuhi syarat tertentu sebagai tabel yang baik. Dekomposisi dapat dikatakan berhasil jika tabel yang dikenal dekomposisi bila digabungkan kembali dapat menjadi tabel awal sebelum di –dekomposisi. Dekomposisi akan sering dilakukan dalam proses normalisasi untuk memenuhi syarat-syaratnya.

2. Bentuk tidak normal, pada bentuk ini semua data yang ada pada tiap entity (diambil atributnya) masih ditampung dalam satu tabel besar. Data yang ada pada tabel ini masih ada yang redundansi dan ada juga yang kosong. Semuanya masih tidak tertata rapi.
3. Normal Form pertama(1st Normal Form), pada tahapan ini tabel di-dekomposisi dari tabel bentuk tidak normal yang kemudian dipisahkan menjadi tabel-tabel kecil yang memiliki kriteria tidak memiliki atribut yang bernilai ganda dan komposit. Semua atribut harus bersifat atomik.
4. Normal Form kedua(2nd Normal Form), pada tahapan ini tabel dianggap memenuhi normal kedua jika pada tabel tersebut semua atribut yang bukan kunci primer bergantung penuh terhadap kunci primer tabel tersebut .
5. Normal Form ketiga(3rd Normal Form), setiap atribut pada tabel selain kunci primer atau kunci utama harus bergantung penuh pada kunci utama. Bentuk normal ketiga biasanya digunakan bila masih ada tabel yang belum efisien. Biasanya penggunaan bentuk normal(normalisasi) hanya sampai pada bentuk ketiga, dan tabel yang dihasilkan telah memiliki kualitas untuk membentuk sebuah database yang dapat diandalkan. Semua tabel diatas juga telah memenuhi bentuk normal tahap ketiga. (Wahana Komputer, 2010 : 32 - 35).

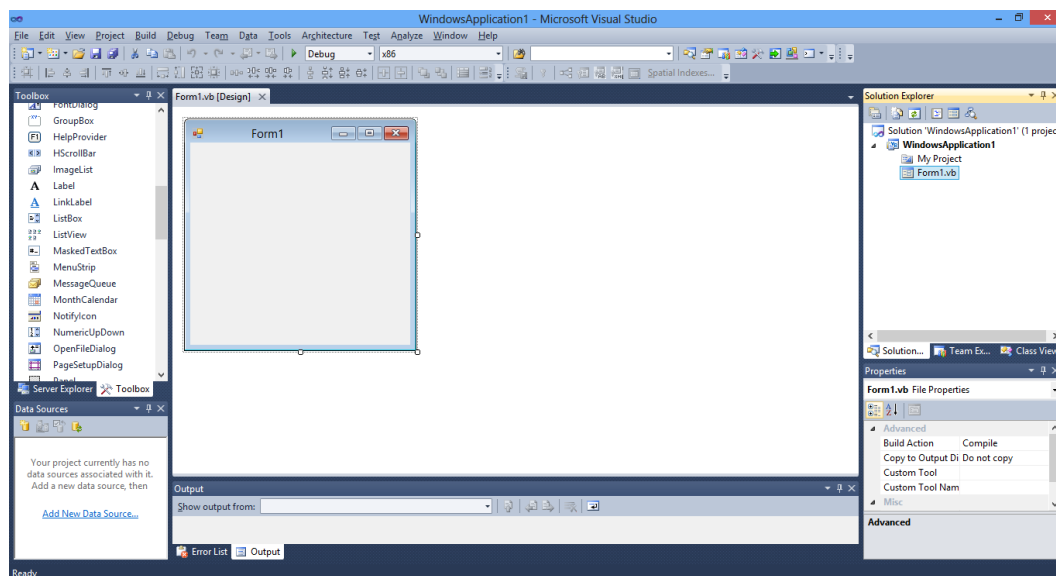
II.9. Pengertian Database

Database atau basis *data* adalah sekumpulan *data* yang memiliki hubungan secara logika dan diatur dengan susunan tertentu serta disimpan dalam media penyimpanan komputer. Data itu sendiri adalah representasi dari semua

fakta yang ada pada dunia nyata. *Database* sering digunakan untuk melakukan proses terhadap data-data tersebut untuk menghasilkan informasi. Dalam *database* ada sebutan-sebutan untuk satuan data yaitu :

1. Karakter, ini adalah satuan data terkecil. *Data* terdiri atas susunan karakter yang pada akhirnya mewakili data yang memiliki arti dari sebuah fakta.
2. *Field*, adalah kumpulan dari karakter yang memiliki fakta tertentu, misalnya seperti nama siswa, tanggal lahir, dan lain-lain.
3. *Record*, adalah kumpulan dari *field*. Pada *record* anda dapat menemukan banyak sekali informasi penting dengan cara mengombinasikan *field-field* yang ada.
4. Tabel, adalah sekumpulan dari *record-record* yang memiliki kesamaan entity dalam dunia nyata. Kumpulan tabel adalah *database*, wujud fisik sebuah *database* dalam komputer adalah sebuah file yang didalamnya terdapat berbagai tingkatan data yang telah disebutkan diatas.
5. *File* adalah bentuk fisik dari penyimpanan data. *File database* berisi semua data yang telah disusun dan diorganisasikan sedemikian rupa sehingga memudahkan pemberian informasi (Wahana Komputer ; 2010 : 24 - 25).

II.10. Sekilas Tentang Visual Basic 2010



Gambar II.4. Tampilan jendela visual basic 2010 setelah *project* disimpan

Visual studio, program yang sedang dikembangkan sering disebut *project* atau *solution*. Disebut *project* karena dibutuhkan lebih dari satu komponen dan satu file untuk membuat sebuah program. Program visual basic 2010 mengandung satu buah file *project* (.vbproj) dan sebuah file solusi (.sln).

Visual Basic.NET merupakan versi visual basic yang mengalami perubahan drastis dan substansial. VB.NET mempertahankan kemudahan dan kesederhanaan dari VB versi sebelumnya ditambah dengan kemampuan berorientasi objek yang mengikuti keandalan C++. Kemampuan berorientasi objek diantaranya mendukung abstraksi, enkapsulasi, inheritance, constructors, polymorphism dan overloading. Dalam VB.NET bisa dibuat class, misalnya pelanggan yang mengabstraksikan pelanggan dalam dunia nyata. Inilah implementasi abstraksi dalam VB.NET.

VB.NET sudah termasuk dalam bahasa pemrograman berorientasi objek, dan bersama dengan bahasa lain dalam visual studio.NET menjadi andalan Microsoft dalam pengembangan bahasa pemrograman berorientasi objek dimasa depan (Julius Hermawan : Tanpa Tahun : 8).

II.11. SQL Server 2008 R2

SQL Server 2008 R2 merupakan DBMS (Database Management System) yang handal dalam mengolah data dengan disertai user interface yang cukup mudah untuk digunakan. Di *SQL Server 2008 R2* ini terdapat fitur baru yaitu :

- a. *Data Compression*
- b. *Change Data Capture*
- c. *Filtered Indexes*
- d. *Table-Valued Parameter*
- e. *Sparse Column*
- f. *Data Type Baru (Date, Time, Filestream)* (Aryo Nugroho dan SmitDev Community : 2010 :1).

II.12. Unified Modeling Language (UML)

Unified Modelling Language adalah bahasa standar yang digunakan untuk menjelaskan dan memvisualisasikan artifak dari proses analisis dan desain berorientasi objek. UML menyediakan standar pada notasi dan diagram yang bias digunakan untuk memodelkan suatu sistem. UML dikembangkan oleh 3 pendekar

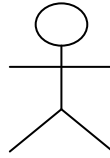
berorientasi objek, yaitu : Grady Booch, Jim Rumbaugh, dan Ivar Jacobson. UML menjadi bahasa yang bias digunakan untuk berkomunikasi dalam perspektif objek antara user dengan developer, antara developer analis dengan developer desain, dan antara developer desain dengan developer pemrograman (Julius Hermawan : Tanpa Tahun : 7).

II.12.1. Notasi - Notasi *UML*

UML menyediakan beberapa notasi dan artifak standar yang bias digunakan sebagai alat komunikasi bagi para pelaku dalam proses analisis dan desain. Artifak dalam UML didefinisikan sebagai informasi dalam berbagai bentuk yang digunakan atau dihasilkan dalam proses pengembangan perangkat lunak. *Use case diagram* merupakan artifak dari proses analisis, sementara *sequence diagram* dan *class diagram* merupakan artifak dari proses desain.

1. Actor

Actor adalah segala sesuatu yang berinteraksi dengan sistem aplikasi komputer. Jadi *actor* ini bias berupa orang, perangkat keras, atau mungkin bias objek lain dalam sistem yang sama. Biasanya yang dilakukan oleh *actor* adalah memberikan informasi pada sistem dan memerintahkan sistem untuk melakukan sesuatu.

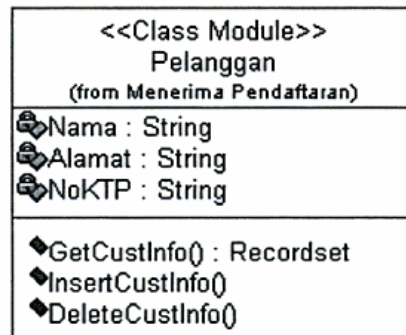


Gambar II.5. Actor

Sumber : Julius Hermawan (Tanpa Tahun : 14)

2. *Class*

Class merupakan pembentuk utama dari sistem berorientasi objek karena *class* menunjukkan kumpulan objek yang memiliki atribut dan operasi yang sama. *Class* digunakan untuk mengimplementasikan interface.



Gambar II.6. Notasi *class*

Sumber : Julius Hermawan (Tanpa Tahun : 14)

Class digunakan untuk mengabstraksikan elemen-elemen dari sistem yang akan dibangun. *Class* bias juga untuk merepresentasikan baik perangkat lunak maupun perangkat keras, baik konsep maupun benda nyata.

3. *Interface*

Interface merupakan sekumpulan operasi tanpa implementasi dari suatu *class*. Implementasi operasi dalam *interface* dijabarkan oleh operasi dalam *class*. Oleh karena itu, keberadaan *interface* selalu disertai oleh *class* yang

mengimplementasikan operasinya. *Interface* ini merupakan salah satu cara mewujudkan prinsip enkapsulasi dalam objek.



Gambar II.7. Notasi *Interface*

Sumber : Julius Hermawan (Tanpa Tahun : 15)

4. *Use Case*

Use case menjelaskan urutan kegiatan yang dilakukan *actor* dan sistem untuk mencapai suatu tujuan tertentu. Walaupun menjelaskan kegiatan namun *use case* hanya menjelaskan apa yang dilakukan oleh *actor* dan sistem, bukan bagaimana *actor* dan sistem melakukan kegiatan tersebut.



Gambar II.8. Notasi *use case*

Sumber : Julius Hermawan (Tanpa Tahun : 16)

Di dalam *use case* terdapat teks untuk menjelaskan urutan kegiatan yang disebut *use case specification*. *Use case specification* terdiri dari :

1. Nama *use case* : mencantumkan nama dari *use case* yang bersangkutan.
Sebaiknya diawali dengan kata kerja untuk menunjukkan suatu aktivitas.
2. Deskripsi singkat (*Brief Description*) : menjelaskan secara singkat dalam satu atau dua kalimat tentang tujuan dari *use case* ini.

3. Aliran normal (*basic flow*) : ini adalah jantung dari *use case*. Menjelaskan interaksi antara *actor* dan sistem dalam kondisi normal, yaitu segala sesuatu berjalan dengan lancar, tiada halangan atau hambatan dalam mencapai tujuan dari *use case*.
4. Aliran alternative (*alternative flow*) : merupakan pelengkap dari dari *basic flow* karena tidak ada yang sempurna dalam setiap kali *use case* berlangsung. Didalam *alternate flow* ini dijelaskan apa yang akan terjadi bila suatu halangan atau hambatan terjadi sewaktu *use case* ini berlangsung. Ini terutama berhubungan dengan error yang mungkin terjadi, misalnya karena sistem kekurangan data untuk diolah, terjadi masalah internal sistem computer, terjadi masalah eksternal.
5. *Special requirement* : berisi kebutuhan lain yang belum tercakup dalam aliran normal dan alternatif. Biasanya secara tegas dibedakan bahwa *basic flow* dan *alternate flow* menangani kebutuhan fungsional dari *use case*. Sementara *special requirement* yang tidak berhubungan dengan kebutuhan fungsional, misalnya kecepatan transaksi maksimum berapa cepat dan berapa lama, kapasitas akses yaitu jumlah *user* yang akan mengakses dalam waktu bersamaan.
6. *Pre-Condition* : menjelaskan persyaratan yang harus dipenuhi sebelum *use case* bisa dimulai.
7. *Post-Condition* : menjelaskan kondisi yang berubah atau terjadi saat *use case* selesai dieksekusi.

5. *Interaction*

Digunakan untuk menunjukkan baik aliran pesan atau informasi antar objek maupun hubungan antar objek. Biasanya *interaction* ini dilengkapi juga dengan teks bernama *operation signature* yang tersusun dari nama operasi, parameter yang terkirim dan tipe parameter yang dikembalikan.

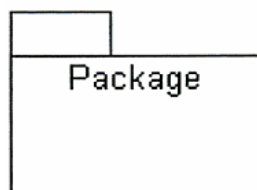


Gambar II.9. Notasi *interaction*

Sumber : Julius Hermawan (Tanpa Tahun : 18)

6. *Package*

Adalah kontainer atau wadah konseptual yang digunakan untuk mengelompokkan elemen-elemen dari sistem yang sedang dibangun, sehingga bisa dibuat model yang lebih sederhana. Tujuannya adalah untuk mempermudah penglihatan dari model yang sedang dibangun.

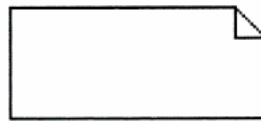


Gambar II.10. Notasi *package*

Sumber : Julius Hermawan (Tanpa Tahun : 19)

7. *Note*

Digunakan untuk memberikan keterangan dan komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model. *Note* ini bisa ditampilkan kesemua elemen notasi yang lain.

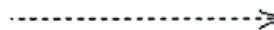


Gambar II.11. Notasi *note*

Sumber : Julius Hermawan (Tanpa Tahun : 18)

8. *Dependency*

Merupakan relasi yang menunjukkan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. Elemen yang ada dibagian tanda panah adalah elemen yang tergantung pada elemen yang ada dibagian tanpa tanda panah.



Gambar II.12. Notasi *dependency*

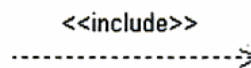
Sumber : Julius Hermawan (Tanpa Tahun : 20)

Terdapat 2 *stereotype* dari *dependency*, yaitu *include* dan *extend*. *Include* menunjukkan bahwa suatu bagian dari elemen (yang ada digaris tanpa panah) memicu eksekusi bagian dari elemen lain (yang ada digaris dengan panah),

misalnya untuk notasi $A \rightarrow B$ operasi yang ada di class A memicu dieksekusinya operasi yang ada di class B.

Extend menunjukkan bahwa suatu bagian dari elemen digaris tanpa panah bisa disisipkan kedalam elemen yang ada di garis dengan panah, misalnya untuk notasi $A \rightarrow B$ suatu fungsi dari *use case* A bisa disisipkan kedalam *use case* B, atau dengan kata lain A optional untuk B.

Kedua *stereotype* ini direpresentasikan dengan menambahkan text include atau extend di notasi *dependency*.



Gambar II.13. Notasi include dependency

Sumber : Julius Hermawan (Tanpa Tahun : 20)

9. Association

Menggambarakan navigasi antar *class* (*navigation*), berapa banyak objek lain yang bisa berhubungan dengan satu objek (*multiplicity* antar *class*), dan apakah suatu *class* menjadi bagian dari *class* lainnya (*aggregation*).

Navigation dilambangkan dengan penambahan tanda panah diakhir garis. *Biridirectional navigation* menunjukkan bahwa dengan mengetahui salah satu *class* bisa didapatkan informasi dari *class* lainnya. Sementara dengan *UniDirectional navigation* hanya dengan mengetahui *class* diujung garis *association* tanpa panah kita bisa mendapatkan informasi dari *class* diujung dengan panah, tetapi tidak sebaliknya.

Multiplicity dinotasikan dengan menambahkan teks (1), (0..1), (0..*), (1..*) atau bilangan tertentu (3) dimasing-masing ujung garis *association*. Misalnya terdapat hubungan sebagai berikut : A(1..*) --- (0..*)B. Cara membacanya adalah bahwa untuk setiap objek B harus berhubungan dengan lebih dari satu objek A atau untuk setiap objek A bisa berhubungan dengan banyak objek B atau tidak sama sekali.

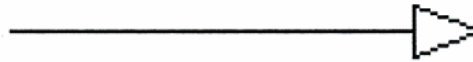
Gambar II.14. Notasi *association*

Sumber : Julius Hermawan (Tanpa Tahun : 20)

10. Generalization

Menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization*, *class* yang lebih spesifik (*subclass*) akan menurunkan atribut dan operasi dari *class* yang lebih umum (*super class*).

Contohnya adalah hubungan antara *class* pelanggan dengan *class* pelanggan antar. *Class* pelanggan adalah abstraksi dari pelanggan secara umum, sementara *class* pelanggan antar mengabstraksikan pelanggan yang memesan lewat telepon dan kaset VCD yang dipesan diantar oleh rental. *Class* pelanggan antar akan memiliki atribut dan operasi *class* pelanggan dengan tambahan atribut bernama biaya antar.

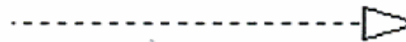


Gambar II.14. Notasi *association*

Sumber : Julius Hermawan (Tanpa Tahun : 22)

11. *Realization*

Menunjukkan hubungan bahwa elemen yang ada di bagian tanpa tanda panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada dibagian panah. Misalnya *class* merealisasikan *package*, *component* merealisasikan *class* atau *interface*.



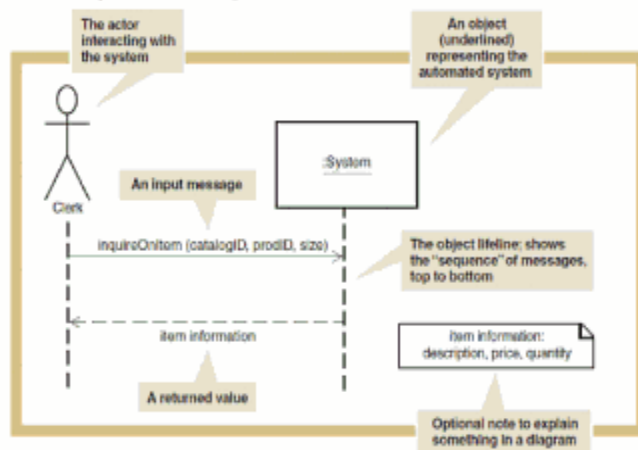
Gambar II.14. Notasi *realization*

Sumber : Julius Hermawan (Tanpa Tahun : 22)

12. *Sequence diagram*

Menurut (John Satzinger : 2010) dalam bukunya “*System Analysis and Design in a Changing World*”, *sequence diagram* adalah diagram yang digunakan untuk mendefinisikan *input* dan *output* serta urutan interaksi antara pengguna dan sistem dalam sebuah *use case* (Evi Triandini dan I Gede Suardika : 2012 : 71).

Notasi Sequence Diagram

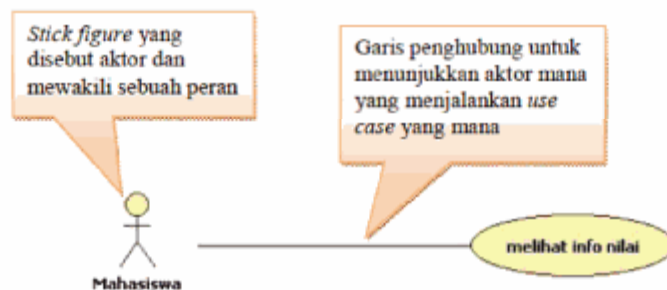


Gambar II.15. Notasi sequence diagram

Sumber : Evi Triandini dan I Gede Suardika (2012 : 71)

13. Use case diagram

Menurut (John Satzinger : 2010) dalam bukunya “*System Analysis and Design in a Changing World*” Use case diagram adalah sebuah kegiatan yang dilakukan oleh sistem, biasanya dalam menanggapi permintaan pengguna sistem (Evi Triandini dan I Gede Suardika : 2012 : 17).

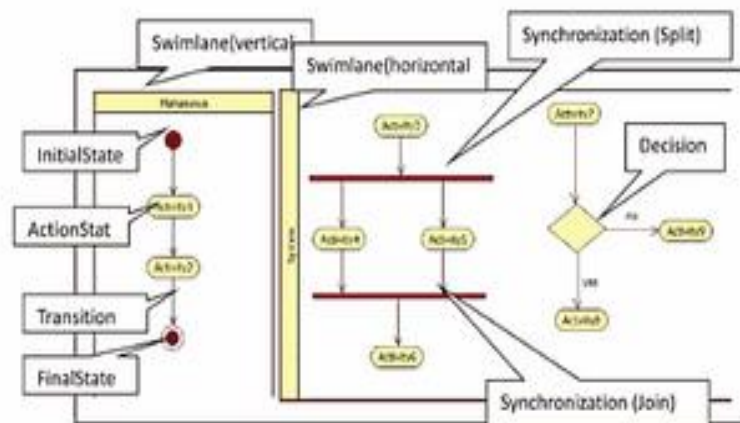


Gambar II.16. Use case diagram

Sumber : Evi Triandini dan I Gede Suardika (2012 : 17)

14 . *Activity diagram*

Menurut (John Satzinger : 2010) dalam bukunya “*System Analysis and Design in a Changing World*” Activity diagram adalah sebuah diagram alur kerja yang menjelaskan berbagai kegiatan pengguna (atau sistem). Orang yang melakukan masing-masing aktivitas, dan aliran sekuensi dan aktivitas-aktivitas tersebut (Evi Triandini dan I Gede Suardika : 2012 : 37).



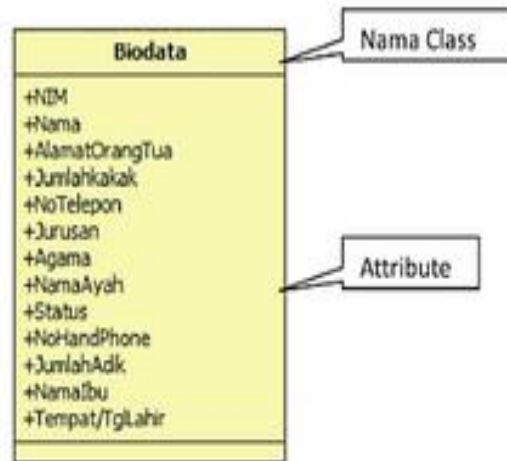
Gambar II.17. Notasi activity diagram

Sumber : Evi Triandini dan I Gede Suardika (2012 : 37)

15. *Class diagram*

Menurut (John Satzinger : 2010) dalam bukunya “*System Analysis and Design in a Changing World*” Ada dua jenis class diagram, yaitu domain class diagram dan design class diagram.

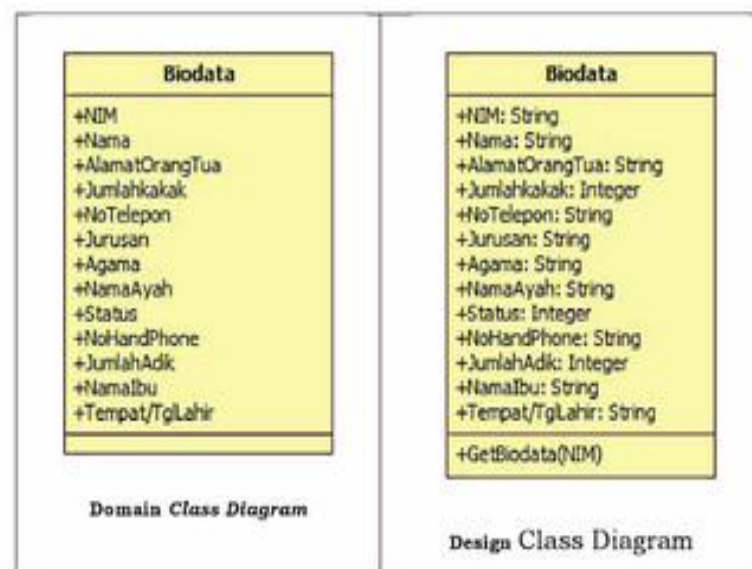
Domain class diagram bertujuan untuk mendokumentasikan dan menggambarkan kelas-kelas dalam pemrograman yang nantinya akan dibangun.



Gambar II.18. Notasi Class Diagram

Sumber : Evi Triandini dan I Gede Suardika (2012 : 49)

Desain class diagram menggambarkan kelas berorientasi objek yang dibutuhkan dalam pemrograman, navigasi diantara kelas, *attributes names*, dan propertinya (Evi Triandini dan I Gede Suardika : 2012 : 49-50).



Gambar II.19. Notasi Class Diagram

Sumber : Evi Triandini dan I Gede Suardika (2012 : 37)