BAB II

LANDASAN TEORI

II.1. Sistem

II.1.1. Konsep Dasar Sistem

Sistem merupakan kumpulan dari unsur atau elemen-elemen yang saling berkaitan/berinteraksi dan saling mempengaruhi dalam melakukan kegiatan bersama untuk mencapai suatu tujuan tertentu (Asbon Hendra, 2011 : 157).

Berikut ini merupakan pengertian sistem dari beberapa ahli:

1. Jerry FithGerald “Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan dan berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu.
2. Ludwig Von Bartalanfy “Sistem merupakan seperangkat unsur yang saling terikat dalam suatu antar relasi di antara unsur-unsur tersebut dengan lingkungan.
3. Anatol Raporot “Sistem adalah suatu kumpulan kesatuan dan perangkat hubungan satu sama lain.
4. L. Ackof “Sistem adalah setiap kesatuan secara konseptual atau fisik yang terdiri dari bagian-bagian dalam keadaan saling tergantung satu sama lainnya.

Dari uraian di atas, sehingga dapat disimpulkan bahwa sistem adalah sekumpulan elemen yang saling terkait atau terpadu untuk mencapai tujuan tertentu.

II.1.2. Pengertian Sistem

Mempelajari suatu sistem akan lebih mengena bila mengetahui terlebih dahulu apakah sistem itu. Pengertian tentang sistem pertama kali dapat diperoleh dari definisi sistem itu sendiri. Jika kita perhatikan dengan seksama, diri kita juga terdiri dari berbagai sistem yang berfungsi untuk mengantar kita pada tujuan hidup kita. (Tata Sutabri, 2012 : 4).

Gordon B. Davis dalam bukunya menyatakan bahwa sistem bisa berupa abstrak atau fisik. Sistem yang abstrak adalah susunan gagasan-gagasan atau konsepsi yang teratur yang saling bergantung. Misalnya, sistem teologi adalah susunan yang teratur dari gagasan-gagasan tentang Tuhan, manusia, dan lain sebagainya. Sedangkan sistem yang bersifat fisik adalah serangkaian unsur yang bekerja sama untuk mencapai suatu tujuan. (Tata Sutabri, 2012 : 6).

II.1.3. Karakteristik Sistem

Model umum sebuah sistem terdiri dari input, proses dan output. Hal ini merupakan konsep sebuah sistem yang sangat sederhana mengigat sebuah sistem dapat mempunyai beberapa masukan dan keluaran sekaligus. Selain itu sebuah sistem juga memiliki karakteristik atau sifat-sifat tertentu, yang mencirikan bahwa hal tersebut bisa dikatakan sebagai suatu sistem. (Tata Sutabri, 2012 : 13-14).

Adapun karakteristik yang dimaksud adalah sebagai berikut:

1. Komponen Sistem *(Components)*

   Suatu sistem terdiri dari sejumlah komponen yang berinteraksi, yang bekerja sama membentuk satu kesatuan. Komponen-komponen sistem tersebut dapat berupa suatu bentuk subsistem. Setiap subsistem memiliki sifat-sifat sistem yang menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai sistem yang lebih besar yang disebut dengan supra sistem.

2. Batasan Sistem *(Boundary)*

   Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem lainnya atau sistem dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan yang tidak dapat dipisah-pisahkan.

3. Lingkungan Luar Sistem *(Evirontment)*

   Bentuk apapun yang ada di luar lingkup atau batasan sistem yang mempengaruhi operasi sistem tersebut disebut dengan lingkungan luar sistem. Lingkungan luar sistem ini dapat menguntungkan dan dapat juga merugikan sistem tersebut. Lingkungan luar yang menguntungkan merupakan energi bagi sistem tersebut, yang dengan demikian lingkungan luar tersebut harus selalu dijaga dan dipelihara. Sedangkan lingkungan luar yang merugikan harus dikendalikan. Kalau tidak maka akan mengganggu kelangsungan hidup sistem tersebut

4. Penghubung Sistem *(Interface)*

   Media yang menghubungkan sistem dengan subsistem yang lain disebut dengan penghubung sistem *interface*. Penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem yang lain. Keluaran suatu subsistem akan menjadi masukan untuk subsistem yang lain dengan melewati penghubung. Dengan demikian terjadi suatu integrasi sistem yang membentuk satu kesatuan.

5. Masukan Sistem *(Input)*

   Energi yang dimasukkan ke dalam sistem tersebut masukan sistem, yang dapat berupa pemeliharaan *(maintenance input)* dan sinyal *(signal input)*. Sebagai contoh di dalam suatu unit sistem komputer, "program" adalah *maintenance input* yang digunakan untuk mengoperasikan computer. Sementara "data" adalah signal yang akan diolah menjadi informasi.

6. Keluaran Sistem *(Output)*

   Hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna. Keluaran ini merupakan bagi subsistem yang lain. Seperti contoh sistem informasi, keluaran yang dihasilkan adalah informasi, di mana informasi ini dapat digunakan sebagai masukan untuk pengambilan keputusan atau hal-hal lain yang merupakan input bagi subsistem lainnya.

7. Pengolahan Sistem *(Procces)*

   Suatu sistem dapat mempunyai suatu proses yang akan mengubah masukan menjadi keluaran. Sebagai contoh, sistem akuntansi, Sistem ini akan mengolah data transaksi menjadi laporan-laporan yang dibutuhkan oleh pihak manajemen.

8. Sasaran Sistem *(Objective)*

   Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat deterministik. Kalau suatu sistem tidak memiliki sasaran, maka operasi sistem tidak ada gunanya. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuan yang telah direncanakan.

II.2. Sistem Pendukung Keputusan

Sistem pendukung keputusan adalah sebuah sistem berbasis komputer dengan antar muka antara mesin/komputer dengan pengguna. Sistem pendukung keputusan ditujukan untuk membantu dalam menyelesaikan suatu masalah dalam berbagai level manajemen dan bukan untuk mengganti posisi manusia sebagai pembuat keputusan. Sistem pendukung keputusan mampu memberi alternatif solusi bagi masalah semi/tidak tersetruktur baik bagi perseorangan atau kelompok dan dalam berbagai macam proses dan gaya pengambilan keputusan sistem pendukung keputusan menggunakan data, basis data, dan analisa model-model keputusan. Sistem pendukung keputusan bersifat adaptif, efektif, interaktif, *easy to use* dan *fleksibel* serta menyediakan akses terhadap berbagai macam format dan tipe sumber data (data *source*). (Kardiaman, 2014 : 64).

II.3. Metode *Profile Matching*

Profile Matching merupakan suatu proses yang sangat penting dalam manajemen SDM dimana terlebih dahulu ditentukan kompetensi (kemampuan) yang diperlukan oleh suatu jabatan. Kompetensi/kemampuan tersebut haruslah dapat dipenuhi oleh pemegang/calon pemegang jabatan.

Dalam proses *Profile Matching* secara garis besar merupakan proses membandingkan antara kompetensi individu kedalam kompetensi jabatan sehingga dapat diketahui perbedaan kompetensinya (disebut juga gap), semakin kecil gap yang dihasilkan maka bobot nilainya semakin besar yang berarti memiliki peluang lebih besar untuk karyawan menempati posisi tersebut. (Safrianto, 2015 : 61).

II.3.1. Proses Perhitungan Metode *Profile Matching*

Menurut (Nina Sherly : 2013 ; 44-45) adapun langkah-langkah dalam penyelesaian metode *Profile Matching* adalah sebagai berikut:

1. Perhitungan *GAP* Kompetensi

   Setelah proses pemilihan kandidat, pada proses ini ialah perhitungan pemetaan *gap* kompetensi dimana yang simaksud dengan *gap* adalah beda antara profil jabatan dengan profil karyawan atau dapat ditunjukkan pada rumus di bawah ini:

   Gap = Profil Karyawan – Profil Jabatan

2. Perhitungan Pemetaan *GAP*

Kompetensi berdasarkan aspek-aspek untuk pengumpulan *gap-gap* yang terjadi itu sendiri pada tiap aspeknya mempunyai perhitungan yang berbeda-beda. Setelah didapatkan tiap *gap* masing-masing karywan maka tiap profil karyawan diberi bobot nilai dengan patokan tabel nilai *gap* seperti yang dapat di lihat pada Tabel II.1. di bawah ini:

Tabel II.1. Keterangan Bobot Nilai *Gap*

No.	Selisih	Bobot Nilai	Keterangan
1	0	5	Tidak ada selisih (kompetensi sesuai dengan yang dibutuhkan).
2	1	4,5	Kompetensi individu kelebihan 1 tingkat/level.
3	-1	4	Kompetensi individu kekurangan 1 tingkat/level.
4	2	3,5	Kompetensi individu kelebihan 2 tingkat/level.
5	-2	3	Kompetensi individu kekurangan 2 tingkat/level.
6	3	2,5	Kompetensi individu kelebihan 3 tingkat/level.
7	-3	2	Kompetensi individu kekurangan 3 tingkat/level.
8	4	1,5	Kompetensi individu kelebihan 4 tingkat/level.
9	-4	1	Kompetensi individu kekurangan 4 tingkat/level.

(Sumber : Nina Sherly ; 2013)

3. Perhitungan dan Pengelompokan *Core* dan *Secondary Factor*

Setelah menentukan bobot nilai *gap* untuk setiap kriteria, kemudian tiap aspek kriteria di kelompokkan menjadi 2 (dua) kelompok yaitu

kelompok *Core Factor* dan *Secondary Factor*. Untuk perhitungan *Core Factor* ditunjukkan dengan persamaan (1).

$$\mathbf{NCF} = \frac{\mathbf{\Sigma\, NC(i,s,p)}}{\sum \boldsymbol{IC}} \quad (1)$$

Keterangan:

NCF : Nilai rata-rata *core factor*

NC(i, s, p) : Jumlah total nilai *core factor*

IC : Jumlah *item core factor*

Sedangkan untuk perhitungan *secondary factor* ditunjukkan dengan persamaan (2).

$$\mathbf{NCS} = \frac{\mathbf{\Sigma\, NS(i,s,p)}}{\sum \boldsymbol{IS}} \quad (2)$$

Keterangan:

NSF : Nilai rata-rata *secondary factor*

NS(i, s, p) : Jumlah total nilai *secondary factor*

ICS : Jumlah *item secondary factor*

4. Perhitungan Nilai Total

Dari hasil perhitungan dari tiap aspek di atas kemudian dihitung nilai total berdasarkan presentasi dari *core* dan *secondary factor* yang diperkirakan berpengaruh terhadap kinerja tiap-tiap profil. Contoh perhitungan ditunjukkan dengan persamaan (3).

$$\boldsymbol{N\,(i,s,p) = (x)\%NCF(i,s,p)(x)\%NSF(i,s,p)} \quad (3)$$

Keterangan:

(i,s,p) : (Intelektual, Sikap Kerja, Perilaku)

N(i,s,p) : Nilai rata-rata *core factor*

NCF (i,s,p) : Nilai rata-rata *secondary factor*

(x)% : Nilai persen yang diinputkan

5. Perhitungan Penentuan Hasil Akhir Atau Ranking

Hasil akhir dari proses ini adalah ranking dari kandidat yang diajukan. Adapun perhitungan tersebut ditunjukkan dengan persamaan (4).

$$\boldsymbol{Ha} = (\boldsymbol{x})\%\boldsymbol{Ni} + (\boldsymbol{x})\%\boldsymbol{Ns} + (\boldsymbol{x})\%\boldsymbol{Np} \quad (4)$$

Keterangan:

Ha : Hasil akhir

Ni : Nilai Intelektual

Ns : Nilai Sikap

Np : Nilai Perilaku

(x)% : Nilai persen yang diinputkan

II.4. Pengertian Basis Data

Sebuah basis data adalah sebuah kumpulan data yang saling berhubungan secara logis dan merupakan sebuah penjelasan dari data tersebut, yang didesain untuk menemukan data yang dibutuhkan oleh sebuah organisasi. Basis data juga merupakan sekumpulan data elemen data yang saling terintegrasi (Indrajani ; 2014 : 70).

Berdasarkan tingkat kompleksitas nilai data, tingkatan data dapat disusun dalam sebuah hierarki, mulai dari yang paling sederhana hingga paling sederhana hingga paling kompleks (Edy Sutanta, 2011 : 35-36).

1. Sisem basis data, merupakan sekumpulan subsistem yang terdiri atas basis data dengan para pemakai yang menggunakan basis data secara bersama-sama, personal-personal yang merancang dan mengelola basis data, teknik-teknik untuk merancang dan mengelola basis data, serta sistem komputer untuk mendukungnya.
2. Basis data, merupakan sekumpulan dari bermacam-macam tipe *record* yang memiliki hubungan antar-*record* dan rincian data terhadap obyek tertentu.
3. File, merupakan sekumpulan *record* sejenis secara relasi yang tersimpan dalam media penyimpanan sekunder.
4. *Record,* merupakan *field/*atribut/data item yang saling berhubungan terhadap obyek tertentu.
5. *Data item/field/*atribut, merupakan unit terkecil yang disebut data, sekumpulan *byte* yang mempunyai makna.
6. *Data agregate,* merupakan sekumpulan data item/*field*/atribut dengan ciri tertentu dan diberi nama.
7. *Byte,* adalah bagian terkecil yang dialamatkan dalam memori. *Byte* merupakan sekumpulan *bit* yang secara konvensional terdiri atas kombinasi 8 *bit* biner yang menyatakan sebuah karakter dalam memori (1 *byte* = 1 karakter).
8. *Bit,* adalah sistem biner yang terdiri atas dua macam nilai, yaitu 0 dan 1. Sistem biner merupakan dasar yang dapat digunakan untuk komunikasi antara manusia dan mesin (komputer).

II.5. Kamus Data

Kamus data adalah katalog fakta tentang data dan kebutuhan informasi suatu sistem informasi. Kamus data terdapat pada tahapan analisis dan perancangan. Pada tahap analisis, kamus data berfungsi untuk mendifinisikan data yang mengalir pada sistem. Sedangkan pada tahap perancangan, kamus data ini digunakan untuk merancang masukan dan keluaran seperti laporan serta basis data. Pada DFD aliran data memiliki sifat global, sedangkan pada kamus data dibuat berdasarkan aliran data yang terdapat pada DFD. (Indrajani, 2015 : 30).

Tabel II.2. Notasi Kamus Data

Notasi	Keterangan
=	Is Composed Of
+	And
( )	Optional (May be present or absent)
{ }	Iteration
[ ]	Select one of several alternative choices
**	Comment
@	Identifier (key field) for a store
|	Separates alternative choices in the construct

(Sumber : Indrajani ; 31)

Contoh kamus data, antara lain:

name = courtesy-title + first-name +(middle-name) + last-name

courtesy-title = [Mr. | Miss | Mrs. | Ms. | Dr. | Profesor]

first-name = {legal-character}

middle-name = {legal-character}

last-name = {legal-character}

legal-character = [A-Z|a-z|0-9|'|-| |]

II.6. Normalisasi

Normalisasi diartikan sebagai suatu teknik yang menstrukurkan/mendekomposisi data dalam cara-cara tertentu untuk mencegah timbulnya permasalahan pengolahan data dalam basis data. Permasalahan yang dimaksud adalah berkaitan dengan penyimpangan-penyimpangan *(anomalies)* yang terjadi akibat adanya kerangkapan data dalam relasi dan in-efisiensi pengolahan (Martin, 1975). (Edy Sutanta ; 2011 : 174)

Proses normalisasi menghasilkan relasi yang optimal, yaitu (Martin, 1975) : (Edy Sutanta ; 2011 : 175)

1. Memiliki struktur *record* yang konsisten secara logik;
2. Memiliki struktur *record* yang mudah untuk dimengerti;
3. Memiliki struktur *record* yang sederhana dalam pemeliharaan;
4. Memiliki struktur *record* yang mudah ditampilkan kembali untuk memenuhi kebutuhan pengguna;
5. Minimalisasi kerangkapan data guna meningkatkan kinerja sistem.

Secara berturut-turut masing-masing level normal tersebut dibahas berikut ini, dimulai dari bentuk tidak normal. (Edy Sutanta ; 2011 : 176-179)

1. Relasi bentuk tidak normal (*Un Normalized Form* / UNF)

   Relasi-relasi yang dirancang tanpa mengindahkan batasan dalam defisi basis data dan karakteristik *Relational Database Management System* (RDBM) menghasilkan relasi *Un Normalized Form* (UNF).

Bentuk ini harus di hindari dalam perancangan relasi dalam basis data.

Relasi *Un Normalized Form* (UNF) mempunyai kriteria sebagai berikut.

a. Jika relasi mempunyai bentuk *non flat file* (dapat terjadi akibat data disimpan sesuai dengan kedatangannya, tidak memiliki struktur tertentu, terjadi duplikasi atau tidak lengkap)
b. Jika relasi membuat *set atribut* berulang *(non single values)*
c. Jika relasi membuat *atribut non atomic value*

2. Relasi bentuk normal pertama (*First Norm Form* / 1NF)

   Relasi disebut juga *First Norm Form* (1NF) jika memenuhi kriteria sebagai berikut.

   a. Jika seluruh atribut dalam relasi bernilai *atomic (atomic value)*
   b. Jika seluruh atribut dalam relasi bernilai tunggal *(single value)*
   c. Jika relasi tidak memuat set atribut berulang
   d. Jika semua record mempunyai sejumlah atribut yang sama.

Permasalahan dalam *First Norm Form* (1NF) adalah sebagai berikut.

a. Tidak dapat menyisipkan informasi parsial
b. Terhapusnya informasi ketika menghapus sebuah *record*

3. Bentuk normal kedua *(Second Normal Form /* 2NF)

   Relasi disebut sebagai *Second Normal Form (*2NF) jika memenuhi kriteria sebagai berikut

   a. Jika memenuhi kriteria *First Norm Form (*1NF)

b. Jika semua atribut nonkunci *Functional Dependence* (FD) pada *Primary Key* (PK)

Permasalahan dalam *Second Normal Form* / 2NF adalah sebagai berikut

a. Kerangkapan data *(data redundancy)*
b. Pembaharuan yang tidak benar dapat menimbulkan inkonsistensi data *(data inconsistency)*
c. Proses pembaharuan data tidak efisien

Kriteria tersebut mengidentifikasikan bahwa antara atribut dalam *Second Normal Form* masih mungkin mengalami *Third Norm Form*. Selain itu, relasi *Second Normal Form* (2NF) menuntut telah didefinisikan atribut *Primary Key* (PK) dalam relasi. Mengubah relasi *First Norm Form* (1NF) menjadi bentuk *Second Normal Form* (2NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

a. Identifikasikan *Functional Dependence* (FD) relasi *First Norm Form* (1NF)
b. Berdasarkan informasi tersebut, dekomposisi relasi *First Norm Form* (1NF) menjadi relasi-relasi baru sesuai *Functional Dependence* nya. Jika menggunakan diagram maka simpul-simpul yang berada pada puncak diagram ketergantungan data bertindak *Primary Key* (PK) pada relasi baru

4. Bentuk normal ketiga (*Third Norm Form* / 3NF)

   Suatu relasi disebut sebagai *Third Norm Form* jika memenuhi kriteria sebagai berikut.

a. Jika memenuhi kriteria *Second Normal Form* (2NF)

b. Jika setiap atribut nonkunci tidak (TDF) *(Non Transitive Dependeny)* terhadap *Primary Key* (PK)

Permasalahan dalam *Third Norm Form* (3NF) adalah keberadaan penentu yang tidak merupakan bagian dari *Primary Key* (PK) menghasilkan duplikasi rinci data pada atribut yang berfungi sebagai *Foreign Key* (FK) (duplikasi berbeda dengan keterangan data).

Mengubah relasi *Second Normal Form* (2NF) menjadi bentuk *Third Norm Form* (3NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

a. Identifikasi TDF relasi *Second Normal Form* (2NF)

b. Berdasarkan informasi tersebut, dekomposisi relasi *Second Normal Form* (2NF) menjadi relasi-relasi baru sesuai TDF-nya.

5. Bentuk normal *Boyce-Cood (Boyce-Codd Norm Form* / BCNF)

   Bentuk normal *Boyce-Codd Norm Form* (BCNF) dikemukakan oleh R.F. Boyce dan E.F. Codd. Suatu relasi disebut sebagai *Boyce-Codd Norm Form* (BCNF) jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Third Norm Form* (3NF)

   b. Jika semua atribut penentu (determinan) merupakan CK

6. Bentuk normal keempat (*Forth Norm Form* / 4NF)

   Relasi disebut sebagi *Forth Norm Form* (4NF) jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Boyce-Codd Norm Form*.

b. Jika setiap atribut didalamnya tidak mengalami ketergantungan pada banyak nilai.

7. Bentuk normal kelima (*Fifth Norm Form* / 5NF)

   Suatu relasi memenuhi kriteria *Fifth Norm Form* (5NF) jika kerelasian antar data dalam relasi tersebut tidak dapat direkonstruksi dari struktur relasi yang sederhana.

8. Bentuk normal kunci domain (*Domain Key Norm Form* / DKNF)

   Relasi disebut sebagai *Domain Key Norm Form* (DKNF) jika setiap batasan dapat disimpulkan secara sederhana dengan mengetahui sekumpulan nama atribut dan domainnya selama menggunkan sekumpulan atribut pada kuncinya.

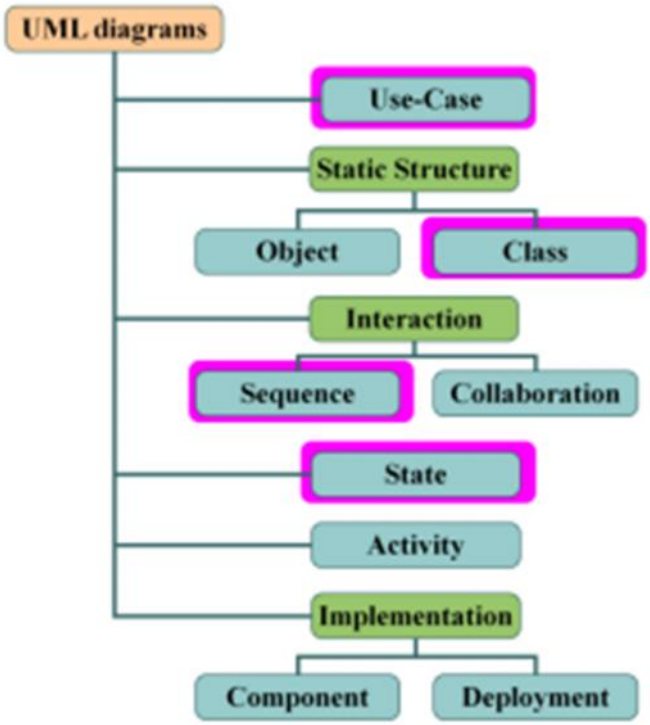
II.7. *Unified Modeling Language* (UML)

UML adalah metode pemodelan berorientasi objek. Sebelum UML telah banyak dibuat pemodelan berorientasi objek seperti: Simula -67, Objective C, C++, Eiffel, dan CLOS. Sesuai dengan namanya, *Unified,* pada dasarnya UML merupakan pemodelan yang generik, Sifatnya yang general memudahkan UML untuk digunakan pada berbagai tipe sistem yang dimodelkan.

Pada level atas, cara pandang dalam UML dapat dibagi menjadi tiga area: *structural clasification, dynamic behavior,* dan model *management*. Klasifikasi struktural menjelaskan suatu objek dalam sistem dan hubungan antar objek tersebut. Pembahasan di dalamnya menyangkut *class, use case*, komponen dan *node. Dynamic behavior* menjelaskan perilaku sistem dari waktu ke waktu.

Termasuk dalam *dynamic*, *behavior view* adalah *state machine view*, *activity view*, dan *interaction view*. Model *management* menjelaskan pengorganisasian model secara hierarkis. (Maman Abdurohman, dkk : 2010 ; 93).

Secara filosofi UML diilhami oleh konsep yang telah ada yaitu konsep permodelan *Object Oriented* karena konsep ini menganalogikan sistem seperti kehidupan nyata yang didominasi oleh Program Studi Ilmu Komputer Universitas Mulawarman obyek dan digambarkan atau dinotasikan dalam simbol-simbol yang cukup spesifik. (Haviluddin, 2011 : 1-2).



Gambar II.1. Diagram UML

(Sumber : Jurnal Informatika Mulawarman)

II.7.1. Tujuan Pemanfaatan *Unified Modeling Language* (UML)

Tujuan dari penggunaan diagram seperti diungkapan oleh Schmuller J. (2004), *"The purpose of the diagrams is to present multiple views of a system; this set of multiple views is called a model"*.

Berikut tujuan utama dalam desain UML adalah (Sugrue J. 2009) : (Haviluddin, 2011 : 2)

1. Menyediakan bagi pengguna (analisis dan desain sistem) suatu bahasa pemodelan visual yang ekspresif sehingga mereka dapat mengembangkan dan melakukan pertukaran model data yang bermakna.
2. Menyediakan mekanisme yang spesialisasi untuk memperluas konsep inti.
3. Karena merupakan bahasa pemodelan visual dalam proses pembangunannya maka UML bersifat independen terhadap bahasa pemrograman tertentu.
4. Memberikan dasar formal untuk pemahaman bahasa pemodelan.
5. Mendorong pertumbuhan pasar terhadap penggunaan alat desain sistem yang berorientasi objek (OO).
6. Mendukung konsep pembangunan tingkat yang lebih tinggi seperti kolaborasi, kerangka, pola dan komponen terhadap suatu sistem.
7. Memiliki integrasi praktik terbaik.

Ada 4 (empat) prinsip dasar dari pemrograman berorientasi obyek yang menjadi dasar kemunculan UML, yaitu *abstraksi, enkapsulasi, modularitas* dan *hirarki*. Berikut dijelaskan satu persatu secara singkat.

1. *Abstraksi* memfokuskan perhatian pada karakteristik obyek yang paling penting dan paling dominan yang bisa digunakan untuk membedakan obyek tersebut dari obyek lainnya.
2. *Enkapsulasi* menyembunyikan banyak hal yang terdapat dalam obyek yang tidak perlu diketahui oleh obyek lain. Dalam praktek pemrograman, enkapsulasi diwujudkan dengan membuat suatu kelas interface yang akan dipanggil oleh obyek lain, sementara didalam obyek yang dipanggil terdapat kelas lain yang mengimplementasikan apa yang terdapat dalam kelas *interface*.
3. *Modularitas* membagi sistem yang rumit menjadi bagian-bagian yang lebih kecil yang bisa mempermudah developer memahami dan mengelola obyek tersebut.
4. *Hirarki* berhubungan dengan abstraksi dan modularitas, yaitu pembagian berdasarkan urutan dan pengelompokkan tertentu. Misalnya untuk menentukan obyek mana yang berada pada kelompok yang sama, obyek mana yang merupakan komponen dari obyek yang memiliki hirarki lebih tinggi. Semakin rendah hirarki obyek berarti semakin jauh abstraksi dilakukan terhadap suatu obyek.

II.7.2. Komponen-komponen *Unified Modeling Language* (UML)

Sejauh ini para pakar merasa lebih mudah dalam menganalisa dan mendesain atau memodelkan suatu sistem karena UML memiliki seperangkat aturan dan notasi dalam bentuk grafis yang cukup spesifik (Sugrue J. 2009).

Komponen atau notasi UML diturunkan dari 3 (tiga) notasi yang telah ada sebelumnya yaitu *Grady Booch, OOD (Object-Oriented Design), Jim Rumbaugh, OMT (Object Modelling Technique),* dan *Ivar Jacobson OOSE (Object-Oriented Software Engineering).*

Pada UML versi 2 terdiri atas tiga kategori, diantaranya : (Haviluddin, 2011 : 3)

1. Struktur Diagram

   Menggambarkan elemen dari spesifikasi dimulai dengan kelas, obyek, dan hubungan mereka, dan beralih ke dokumen arsitektur logis dari suatu sistem. Beberapa struktur diagram dalam UML terdiri atas :

   a. *Class diagram*

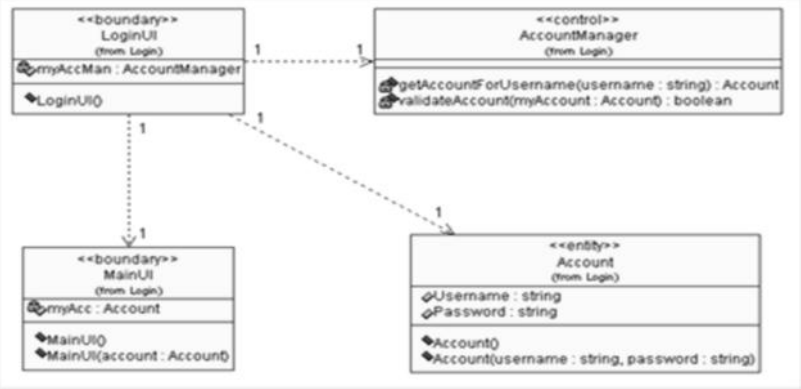
      Class diagram menggambarkan struktur statis dari kelas dalam sistem anda dan menggambarkan atribut, operasi dan hubungan antara kelas. *Class diagram* membantu dalam memvisualisasikan struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak dipakai. Selama tahap desain, *class diagram* berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat.

      Class memiliki tiga area pokok :

1) Nama (dan *stereotype*)

2) Atribut

3) Metoda



Gambar II.2. Notasi Class Diagram
(Sumber : Haviluddin, 2011 : 3)

2. *Behavior Diagram*

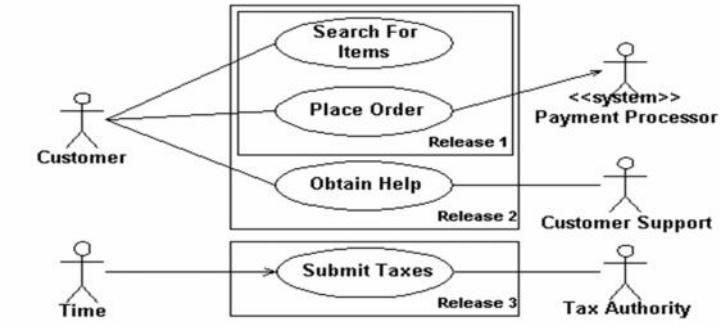
Menggambarkan ciri-ciri behavior/metode/fungsi dari sebuah sistem atau *business process*. *Behavior diagram* dalam UML diantaranya terdiri atas :

a. *Use case diagram*

Diagram yang menggambarkan *actor*, *use case* dan relasinya sebagai suatu urutan tindakan yang memberikan nilai terukur untuk aktor. Sebuah *use case* digambarkan sebagai *elips horizontal* dalam suatu diagram UML *use case*. *Use Case* memiliki dua istilah, yaitu :

1) *System use case*; interaksi dengan sistem.

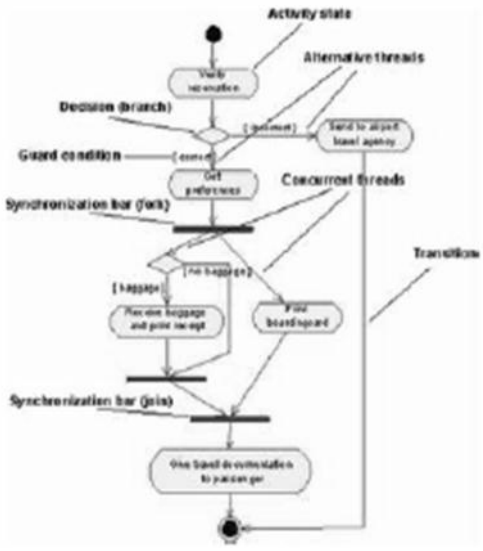
2) *Business use case*; interaksi bisnis dengan konsumen atau kejadian nyata



Gambar II.3. Notasi Use Case Diagram
(Sumber : Haviluddin, 2011 : 4)

b. *Activity diagram*

Menggambarkan aktifitas-aktifitas, objek, *state*, *transisi state* dan *event*. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku sistem untuk aktivitas. Berikut notasi *object diagram* dapat dilihat pada Gambar II.4. di bawah ini.



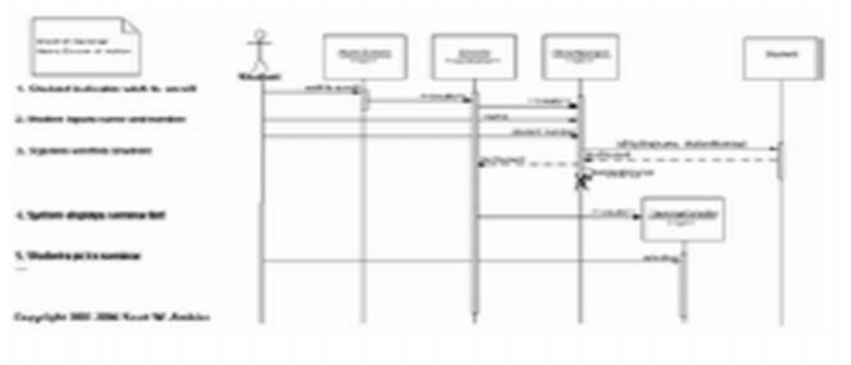
Gambar II.4. Notasi Activity Diagram
(Sumber : Haviluddin, 2011 : 4)

3. *Interaction Diagram*

Bagian dari *behavior diagram* yang menggambarkan interaksi objek. *Interaction diagram* dalam UML salah satunya adalah :

a. *Sequence diagram*

Sequence diagram menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. Secara mudahnya *sequence diagram* adalah gambaran tahap demi tahap, termasuk kronologi (urutan) perubahan secara logis yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case diagram.*



Gambar II.5. Notasi Sequence Diagram
(Sumber : Haviluddin, 2011 : 5)

Untuk menggambarkan analisa dan desain diagram, UML memiliki seperangkat notasi yang akan digunakan ke dalam tiga kategori diatas yaitu struktur diagram, behaviour diagram, dan interaction diagram. Berikut beberapa notasi dalam UML diantaranya :

1) *Actor*, menentukan peran yang dimainkan oleh user atau sistem lain yang berinteraksi dengan subjek. *Actor* adalah segala sesuatu yang berinteraksi langsung dengan sistem aplikasi komputer,

seperti orang, benda atau lainnya. Tugas actor adalah memberikan informasi kepada sistem dan dapat memerintahkan sistem untuk melakukan sesuatu tugas.

2) *Class diagram,* Notasi utama dan yang paling mendasar pada diagram UML adalah notasi untuk mempresentasikan suatu class beserta dengan atribut dan operasinya. Class adalah pembentuk utama dari sistem berorientasi objek.

3) *Use Case* dan *use case specification, Use case* adalah deskripsi fungsi dari sebuah sistem perspektif pengguna. Use case bekerja dengan cara mendeskripsikan tipikal interaksi antara user (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut skenario. Use case merupakan awal yang sangat baik untuk setiap fase pengembangan berbasis objek, design, testing, dan dokumentasi yang menggambarkan kebutuhan sistem dari sudut pandang di luar sistem. Perlu diingat bahwa use case hanya menetapkan apa yang seharusnya dikerjakan oleh sistem, yaitu kebutuhan fungsional sistem dan tidak untuk menentukan kebutuhan nonfungsional, misalnya: sasaran kinerja, bahasa pemrograman dan lain sebagainya.

4) *Interaction, Interaction* digunakan untuk menunjukkan baik aliran pesan atau informasi antar obyek maupun hubungan antar obyek.

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

II.8. *Microsoft Visual Basic 2010*

Visual Basic 2010 merupakan salah satu pemrograman terbaru yang dikeluarkan oleh *Microsoft,* yaitu *Microsoft Visual Studio 2010. Visual Studio* merupakan produk pemrograman andalan dari *Microsoft Corporation*, dimana didalamnya berisi beberapa jenis *IDE* pemrograman seperti *Visual Basic, Visual C++, Visual Web Developer, Visual C# dan Visual F#.*

Semua IDE pemrograman tersebut sudah mendukung penuh implementasi *.Net Framework* terbaru, yaitu *.Net Framework 4.0* yang merupakan pengembangan dari *.Net Framework 3.5*. Adapun database standart yang disertaka adalah *SQL Server 2008 Express*.

Visual Basic 2010 merupakan versi perbaikan dan pengembangan dari versi pendahulunya, yaitu *Visual Basic 2008*. Beberapa pengembangan yang terdapat didalamnya antara lain dukungan terhadap *library* terbaru dari *Microsoft*, yaitu *.Net FrameWork 4.0*, dukungan terhadap aplikasi berbasis *Cloud*

Computing, serta perluasan dukungan terhadap database-database, baik *standalone* maupun *database server*. (Wahana Komputer, 2011 : 2).

II.9. *Microsoft SQL Server 2008*

SQL *(Structured Query Language)* adalah sebuah bahasa yang dipergunakan umtuk mengakses data dalam basis data relasional. Bahasa ini secara *de facto* merupakan bahasa standar yang dipergunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya.

SQL terdiri dari dua bahasa, yaitu *Data Definition Language Manipulation Language (DML)*. Implementasi DDL dan DML sistem manajemen basis data *(SMBD)*, namun secara umum implemen bahasa ini memiliki bentuk standar yang ditetapkan oleh *ANSI*. (Adelia, Jimmy Setiawan : 2011 ; 115).