BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem adalah sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan. Sebagai gambaran, jika dalam sebuah sistem terdapat elemen yang tidak memberikan manfaat dalam mencapai tujuan yang sama, maka elemen tersebut dapat dipastikan bukanlah bagian dari sistem. (Kadir; 2014 ; 61).

II.1.1. Elemen Sistem

Elemen – elemen yang membentuk sebuah sistem yaitu :

a. Tujuan

Setiap sistem memiliki tujuan (*goal*), entah hanya satu atau mungkin banyak. Tujuan inilah yang menjadi pemotivasi yang mengarahkan sistem. Tanpa tujuan, sistem menjadi tidak terarah dan tidak terkendali. Tentu saja, tujuan antara satu sistem dengan sistem lain berbeda – beda. Begitu pula yang berlau pada sistem informasi. Setiap sistem informasi memiliki suatu tujuan, tetapi denga tujuan yang berbeda – beda. Walaupun begitu, tujuan utama yang umum ada tiga macam, yaitu :

1. Untuk mendukung fungsi kepengurusan manajemen,
2. Untuk mendukung pengambilan keputusan manajemen,
3. Untuk mendukung kegiatan operasi perusahaan.

b. Masukan

Masukan (*input*) sistem adalah segala sesuatu yang masuk kedalam sistem dan selanjutnya menjadi bahan untuk diproses. Masukan dapat berupa hal-hal berwujud (tampak secara fisik) maupun yang tidak tampak. Contoh masukan yang berwujud adalah bahan mentah, sedangkan contoh yang tidak berwujud adalah informasi (misalnya permintaan jasa dari pelanggan). Pada sistem informasi, masukan dapat berupa data transaksi, dan data non-transaksi (misalnya, surat pemberitahuan), serta instruksi.

c. Proses

Proses merupakan bagian yang melakukan perubahan atau transformasi dari masukan menjadi keluaran yang berguna, misalnya berupa informasi dan produk, tetapi juga bias berupa hal – hal yang tidak berguna, misalnya saja sisa pembuangan atau limbah. Pada pabrik kimia, proses dapat berupa pemanasan bahan mentah. Pada rumah sakit, proses dapat berupa aktivitas pembedahan pasien. Pada sistem informasi, proses dapat berupa suatu tindakan yang bermacam – macam. Meringkas data, melakukan perhitungan, dan mengurutkan data merupakan beberapa contoh proses.

d. Keluaran

Keluaran (*output*) merupakan hasi dari pemrosesan. Pada sistem informasi, keluaran bias berupa suatu informasi, saran, cetakan laporan, dan sebagainya. (Kadir; 2014 ; 62).

II.2. Informasi

McFadden, dkk. (1999) Mendefenisikan informasi sebagai data yang telah diproses sedemikian rupa sehingga meningkatkan pengetahuan seseorang yang menggunakan data tersebut. Shannon dan Weaver, dua orang insinyur listrik, melakukan pendekatan secara matematis untuk mendefenisikan informasi (Kroenke, 1992). Menurut mereka, informasi adalah "jumlah ketidakpastian yang dikurangi ketika sebuah pesan diterima". Artinya, dengan adanya sistem informasi, tingkat kepastian menjadi meningkat. Menurut Davis (1999), informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau saat mendatang. (Kadir; 2014 ; 45).

II.3. Sistem Informasi

Ada beragam defenisi sistem informasi, sebagaimana tercantum di Tabel 2.1 Berdasarkan berbagai defenisi tersebut, dapat disimpulkan bahwa sistem informasi mencakup sejumlah komponen (manusia, komputer, teknologi informasi, dan prosedur kerja), ada sesuatu yang diproses (data menjadi informasi), dan dimaksudkan untuk mencapai suatu sasaran atau tujuan. (Kadir; 2014 ; 8).

Tabel II.1 Defenisi Sistem Informasi

Sumber	Defenisi
Alter (1992)	Sistem informasi adalah kombinasi antar prosedur kerja, informasi, orang, dan teknologi informasi yang diorganisasikan untuk mencaai tujuan dalam sebuah organisasi.
Bodnar dan Hopwood (1993)	Sistem informasi adalah sekumpulan perangkat keras dan perangkat lunak yang dirancang untuk mentransformasikan data kedalam bentuk informasi yang berguna.
Genilas, Oram, dan Wiggins (1990)	Sistem informasi adalah suatu sistem buatan manusia yang secara umum terdiri atas sekumpulan komponen berbasis computer dan manual yang dibuat untuk menghimpun, menyimpan, dan mengelola data serta menyediakan informasi keluaran kepada para pemakai.
Hall (2001)	Sistem informasi adalah sebuah rangkaian prosedur formal dimana data dikelompokkan, diproses menjadi informasi, dan didistribusikan kepada pemakai.
Turban, McLean, dan Wetherbe (1999)	Sebuah sistem informasi mengumpulkan, memproses, menyimpan, menganalisis, dan menyebarkan informasi untuk tujuan yang spesifik.
Wilkinson (1992)	Sistem informasi adalah kerangka kerja yang mengoordinasikan sumber daya (manusia, komputer) untuk mengubah masukan (*input*) menjadi keluaran (informasi), guna mencapai sasaran – sasaran perusahaan.

(Sumber : Kadir ; 2014 ; 8)

II.3.1. Komponen Sistem Informasi

Sistem informasi mengandung komponen-komponen seperti berikut.

- Perangkat keras (*hardware)*, yang mencakup peranti-peranti fisik seperti komputer dan printer.
- Peraangkat lunak (*software*) atau program, yaitu sekumpulan instruksi yang memungkinkan perangkat keras memproses data.
- Prosedur, yaitu sekumpulan aturan yang dipakai untuk mewujudkan pemrosesan data dan pembangkitan keluaran yang dikehendaki.
- Orang, yakni semua pihak yang bertanggung jawab dalam pengembangan sistem informasi, pemrosesan, dan penggunaan keluaran sistem informasi.
- Basis data (*database*), yaitu kumpulan tabel, hubungan, dan lain-lain yang berkaitan dengan penyimpanan data.
- Jaringan komputer dan komunikasi data, yaitu sistem penghubung yang memungkinkansumber (*resources*) dipakai secara bersama atau diakses oleh sejumlah pemakai. (Kadir; 2014 ; 71).

II.3.2. Klasifikasi Sistem Informasi

Ada berbagai cara untuk mengelompokkan sistem informasi. Klasifikasi yang umum dipakai antara lain :

- Level organisasi
- Area fungsional
- Dukungan yang diberikan, dan
- Arsitektur sistem informasi

Beberapa istilah sistem informasi lain juga sering dijumpai dalam literature, misalnya sistem informasi strategis dan sistem informasi geografis. (Kadir; 2014 ; 89).

II.4. Sistem Pendukung Keputusan

Sistem pendukung keputusan (SPK) atau *Decision Support Systems (DSS)* adalah sistem informasi interaktif yang menyediakan informasi, pemodelan, dan pemanipulasian data yang digunakan untuk membantu pengambilan keputusan pada situasi yang semiterstruktur dan situasi yang tidak terstruktur di mana tek seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat. Konsep *DSS* dikemukakan pertama kali oleh Scott-Morton pada tahun 1971. Beliau mendefenisikan cikal bakal *DSS* tersebut sebagai ”Sistem berbasis komputer interaktif, yang membantu pengambil keputusan menggunakan data dan model untuk memecahkan persoalan-persoalan tidak terstruktur”.

DSS dibuat sebagai reaksi atas ketidakpuasan terhadap TPS dan MIS. Sebagaimana diketahui, TPS lebih memfokuskan diri dari pada perekaman dan pengendalian transaksi yang merupakan kegiatan yang bersifat berulang dan terdefenisi dengan baik, sedangkan MIS lebih berorientasi pada penyediaan laporan bagi manajemen yang sifatnya tidak fleksibel. *DSS* lebih ditujukan untuk mendukung manajemen dalam melakukan pekerjaan yang bersifat analitis, dalam situasi yang kurang terstruktur dan dengan kriteria yang kurang jelas. *DSS* tidak dimaksudkan untuk mengotomasikan pengambilan keputusan, tetapi memberikan perangkat interaktif yang memungkinkan pengambil keputusan dapat melakukan

berbagai analisis dengan menggunakan model-model yang tersedia. (Kadir; 2014 ; 108).

Adapun beberapa karakteristik yang terdapat pada sistem pendukung keputusan adalah sebagai berikut :

- Menawarkan keluasan, kemudahan beradaptasi, dan tanggapan yang cepat.
- Memungkinkan pemakai memulai dan mengendalikan masukan dan keluaran.
- Dapat dioperasikan dengan sedikit atau tanpa bantuan pemrogram professional.
- Menyediakan dukungan untuk keputusan dan permasalahan yang solusinya tak dapat ditentukan di depan.
- Menggunakan analisis data dan perangkat pemodelan yang canggih. (Kadir; 2014 ; 108).

II.5. Metode *SAW*

Metode *SAW* merupakan metode *MDAM* yang paling sederhana dan paling banyak digunakan. Metode ini juga metode yang paling mudah untuk diaplikasikan. Metode *SAW* sering juga dikenal sebagai metode penjumlahan terbobot. Konsep dasar metode adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut. Metode *SAW* membutuhkan proses normalisasi matiks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternatif yang ada. (Oktaputra dan Noersasongko; 2014 ; 3).

Metode *SAW* sering juga dikenal istilah metode penjumlahan terbobot. Konsep dasar metode *SAW* adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut. Metode *SAW* membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternatif yang ada. (Maulan;, 2014; 50).

Keterangan :

rij =nilai rating kinerja ternormalisasi

xij = nilai atribut yang dimiliki dari setiap kriteria

Max xij = nilai terbesar dari setiap kriteria

Min xij = nilai terkecil dari setiap kriteria

benefit = jika nilai terbesar adalah terbaik

cost = jika nilai terkecil adalah terbaik

dimana rij adalah rating kinerja ternormalisasi dari alternatif Ai pada atribut Cj; i=1,2,...,m dan j=1,2,...,n.

Nilai preferensi untuk setiap alternatif (Vi) diberikan sebagai:

$$V_i = \sum_{j=1}^{n} w_j r_{ij}$$

Keterangan :

Vi = rangking untuk setiap alternatif

wj = nilai bobot dari setiap kriteria

rij = nilai rating kinerja ternormalisasi

Nilai Vi yang lebih besar mengindikasikan bahwa alternatif Ai lebih terpilih.

II.2.1. Algoritma Metode *Simple Additive Weight (SAW)*

1. Algoritma Normalisasi

Input : max ij,xij(X=nilai)

Output : rij

Proses :

IF X11>X21>X31>X41 then Max=X11

IF X12>X22>X32>X42 then Max=X12

IF X13>X23>X33>X43 then Max=X13

IF X14>X24>X34>X44 then Max=X14

IF X15>X25>X35>X45 then Max=X15

IF X16> X26>X36>X46 then Max=X16

2. Algoritma preferensi

Input : rij ; W

Ouput : vi

Proses:

V11 =(W1 x X11)+(W2 x X12)+(W3 x X13)+(W4 x X14)

V21 =(W2 x X21)+(W3 x X22)+(W4 x X23)+(W5 x X24)

V31 =(W3 x X31)+(W4 x X32)+(W5 x X33)+(W6 x X34)

V41 =(W4 x X41)+(W5 x X42)+(W6 x X43)+(W7 x X44)

V51 =(W5 x X51)+(W6 x X52)+(W7 x X53)+(W8 x X54)

V61 =(W6 x X61)+(W7 x X62)+(W8 x X63)+(W9 x X64)

Berdasarkan hasil perhitungan preferensi diatas maka yang lebih tinggi nilainya adalah alternatif yang terbaik.

II.6. *Microsoft Visual Basic 2010*

Visual Basic 2010 merupakan salah satu bagian dari produk pemrograman terbaru yang dikeluarkan oleh *Microsoft*, yaitu *Microsoft Visual Studio 2010. Visual Studio* merupakan produk pemrograman andalan dari *mocrosoft corporation*, dimana di dalamnya 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 standar yang disertakan adalah *Microfot SQL Server 2008 express*.

Visual Basic 2010 merupakan versi perbaikan dan pengembangan dari versi pendahulunya yaitu *visual basic 2008*. Beberapa pengembangan yang terdapat di dalamnya antara lain dukungan terhadap *library* terbaru dari *Microsoft*, yaitu *.Net Framework 4.0,* dukungan terhadap pengembangan aplikasi menggunakan *Microsoft SilverLight*, dukungan terhaddap aplikasi berbasis *cloud computing*, serta perluasan dukungan terhadap *database-database*, baik *standalone* maupun *database server.* (Wahana Komputer; 2011 ; 1).

II.7. Basis Data Dan *DBMS*

Basis data dapat didefenisikan sebagai koleksi dari data-data yang terorganisasi sedemikian rupa sehingga data mudah disimpan dan dimanipulasi (diperbarui, dicari, diolah dengan perhitungan-perhitungan tertentu, serta dihapus). Secara teoritis, basis data tidak harus berurusan dengan komputer

(misalnya, catatan belanja hari ini yang dibuat oleh seorang ibu rumah tangga juga merupakan basis data dalam bentuk yang sangat sederhana). (Nugroho; 2011 ; 4).

Menurut Kadir (2014) basis data (*database)* adalah suatu pengorganisasian sekumpulan data yang saling terkait sehingga memudahkan aktifitas untuk memperoleh informasi. Basis data dimaksudkan untuk mengatasi problem pada sistem yang memakai pendekatan berbasis berkas.

Untuk mengelola basis data diperlukan perangkat lunak yang disebut *Database Management System* (*DBMS*). *DBMS* adalah perangkat lunak sistem yang memungkinkan para pemakai membuat, memelihara, mengontrol dan mengakses basis data dengan cara yang praktis dan efisien. *DBMS* dapat digunakan untuk mengakomodasikan berbagai macam pemakai yang memiliki kebutuhan akses yang berbeda-beda. (Kadir; 2014; 218).

Umumnya *DBMS* menyediakan fitur-fitur sebagai berikut :

- Independensi data-program

  Karena basis data ditangani oleh *DBMS*, program dapat dipilih sehingga tidak tergantung pada struktur data dalam basis data. Dengan perkataan lain, program tidak akan terpenaruh sekiranya bentuk fisik data diubah.

- Keamanan

  Keamanan dimaksudkan untuk mencegah pengaksesan data oleh orang yang tidak berwewenang.

- Integritas

Hal ini ditujukan untuk menjaga agar data selalu dalam keadaan yang valid dan konsisten.

- Konkurensi

  Konkurensi memungkinkan data dapat diakses oleh banyak pemakai tanpa menimbulkan masalah.

- Pemulihan (*recovery*)

  DBMS menyediakan mekanisme untuk mengembalikan basis data ke keadaan semula yang konsisten sekiranya terjadi gangguan perangkat keras atau kegagalan perangkat lunak.

- Katalog Sistem

  Katalog Sistem adalah deskripsi tentang data yang terkandung dalam basis data yang dapat diakses oleh pemakai.

- Perangkat Produktivitas

  Untuk menyediakan kemudahan bagi pemakai dan meningkatkan produktivitas, *DBMS* menyediakan sejumlah perangkat produktivitas seperti pembangkit *query* dan pembangkit laporan. (Kadir; 2014 ; 219).

II.8. *SQL Server* 2008

SQL Server 2008 adalah sebuah *RDBMS (Relational Database Management System)* yang sangat powerful dan telah terbukti kekuatannya dalam mengolah data. Dalam versi terbarunya ini, *SQL Server* 2008 memiliki banyak fitur yang bisa dihandalkan untuk meningkatkan performa *database. SQL Server* 2008 memiliki suatu *GUI (Graphic User Interface)* yang kita gunakan untuk melakukan aktivitas sehari hari berkaitan dengan database,seperti menulis *T-SQL*,

melakukan *backup* dan *restore database*, melakukan security database terhadap aplikasi, dan sebagainya. Pada *GUI* tersebut kita bisa melakukan settingan terhadap *SQL Serve*r untuk berkerja lebih optimal.Settingan juga bisa dilakukan menggunakan script untuk memudahkan developer mengubah Setting Opsions pada *SQL Server* 2008. (Ruslan; 2013 ; 39).

II.9. Normalisasi

Normalisasi dapat dipahami sebagai tahapan-tahapan yang masing-masing berhubungan dengan bentuk normal. Bentuk normal adalah keadaan relasi yang dihasilkan dengan menerapkan aturan sederhana berkaitan dengan konsep kebergantungan fungsional pada relasi yang bersangkutan. (Nugroho; 2011 ; 199). Kita akan menggambarkannya secara garis besar sebagai berikut :

1. Bentuk Normal Pertama (1NF/ *First Normal Form*)

   Bentuk normal pertama adalah suatu bentuk relasi dimana atribut bernilai banyak (*multivalues attribute*) telah dihilangkan sehingga kita akan menjumpai nilai tunggal (mungkin saja nilai *null*) pada perpotongan setiap baris dan kolom.

2. Bentuk Normal Kedua (2ND/ *Second Normal Form*)

   Semua kebergantungan fungsional yang bersifat sebagian (*particial functional dependency*) telah dihilangkan.

3. Bentuk Normal Ketiga (3RD/ *Thrid Normal Form*)

   Semua kebergantungan transitif (*transitive dependency*) telah dihilangkan.

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

Semua anomaly yang tersisa dari hasil penyempurnaan kebergantungan fungsional sebelumnya telah dihilangkan.

5. Bentuk Normal Keempat (4NF/ *Fourth Normal Form*)

   Semua kebergantungan bernilai banyak telah dihilangkan.

6. Bentuk Normal Kelima (5NF/ *Fifth Normal Form*)

   Semua anomaly yang tertinggi telah dihilangkan.

II.10. *Unified Modeling Language* (*UML*)

Menurut Windu Gata (2013) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (*UML*). *UML* adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. *UML* saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem. (Urva dan Siregar; 2015 ; 93).

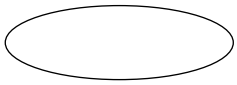
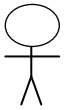
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan *UML* adalah sebagai berikut :

- *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem

informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram dapat dilihat pada tabel II.2 dibawah ini:

Tabel II.2. Simbol *Use Case*

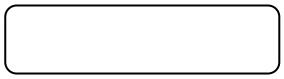
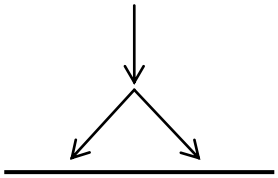
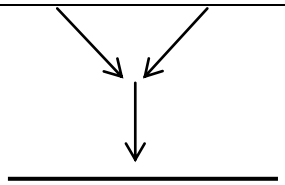
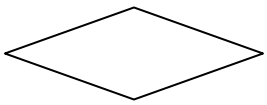
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Urva dan Siregar ; 2015 ; 94)

- Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada tabel II.2 dibawah ini:

Tabel II.3. Simbol *Activity Diagram*

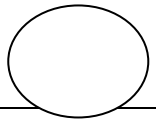
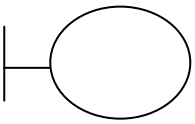
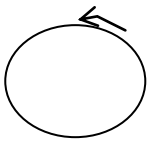
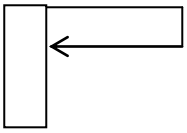
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Urva dan Siregar ; 2015 ; 94)

- Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram* dapat dilihat pada tabel II.4 dibawah ini :

Tabel II.4. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class,* merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class,* berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class,* suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Urva dan Siregar; 2015 ; 95)

- *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.

Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut.

Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti yang dapat dilihat pada tabel II.5 dibawah ini :

Tabel II.5. *Multiplicity Class Diagram*

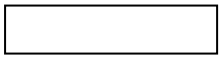
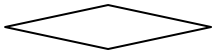
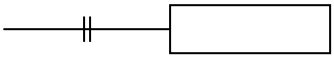
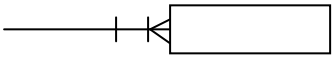
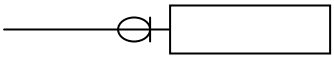
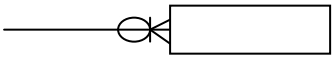
Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Urva dan Siregar; 2015 ; 95)

II.11. *Entity Relationship Diagram*

Entity Relationship Diagram (ERD) adalah bagian yang menunjukkan hubungan antara entity yang ada dalam sistem. Simbol-simbol yang digunakan dapat dilihat dari tabel II.6. (Yuhendra dan Yulianto; 2015 ; 70).

Tabel II.6. Simbol Yang Digunakan Pada *Entity Relationship Diagram (ERD)*

SIMBOL	KETERANGAN
	*Entity*
	Atribut Dan *Entity*
	Atribut Dan *Entity* Dengan *Key* (Kunci)
	Relasi Atau Aktifitas Antar *Entity*
	Hubungan Satu Dan Pasti
	Hubungan Banyak Dan Pasti
	Hubungan Satu Tapi Tidak Pasti
	Hubungan Banyak Tapi Tidak Pasti

(Sumber : Yuhendra dan Yulianto; 2015 ; 70)