

## **BAB II**

### **TINJAUAN PUSTAKA**

#### **II.1. Sistem**

Sistem merupakan sekumpulan elemen-elemen yang saling terintegrasi serta melaksanakan fungsinya masing-masing untuk mencapai tujuan yang telah ditetapkan. Karakteristik sistem terdiri dari :

##### **1. Komponen Sistem**

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk suatu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem.

##### **2. Batasan Sistem**

Batasan merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang suatu kesatuan. Batasan suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

##### **3. Lingkungan Luar Sistem**

Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut.

#### 4. Penghubung Sistem

Penghubung merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lainnya.

#### 5. Masukan Sistem

Masukan sistem adalah energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintenanceinput*) dan masukan sinyal (*signalinput*).

#### 6. Keluaran Sistem

Keluaran sistem adalah hasil energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

#### 7. Pengolah Sistem

Suatu sistem dapat mempunyai suatu bagian pengolah atau sistem itu sendiri sebagai pengolahnya. Pengolah akan mengubah masukan menjadi keluaran.

#### 8. Sasaran Sistem

Suatu sistem mempunyai tujuan (*goal*) atau sasaran (*objective*). Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak ada gunanya (Sulindawati ; 2010 : 135).

### **II.2. Sistem Pendukung Keputusan**

Sistem pendukung keputusan adalah sistem berbasis komputer yang membantu para pengambil keputusan mengatasi berbagai masalah melalui interaksi Pengambilan keputusan (*Decision Making*) adalah melakukan penilaian

dan menjatuhkan pilihan. Keputusan ini diambil setelah melalui beberapa perhitungan dan pertimbangan alternatif. Decision support system atau sistem pendukung keputusan secara umum didefinisikan sebagai sebuah sistem yang mampu memberikan kemampuan baik kemampuan pemecahan masalah maupun kemampuan pengkomunikasian untuk masalah semi terstruktur. Secara khusus, Sistem pendukung keputusan didefinisikan sebagai sebuah sistem yang mendukung kerja seorang manager maupun sekelompok manager dalam memecahkan masalah semi terstruktur dengan cara memberikan informasi ataupun usulan menuju pada keputusan tertentu (Muhammad Yudin Ritonga ; 2014 : 19).

### **II.2.1. Karakteristik Sistem Pendukung Keputusan**

Karakteristik dan kapabilitas sistem pendukung keputusan antara lain :

1. Dukungan untuk pengambilan keputusan, terutama pada situasi semi terstruktur dan tak terstruktur, dengan menyertakan penilaian manusia dan informasi terkomputerisasi. Masalah-masalah tersebut tidak dapat dipecahkan oleh sistem komputer lain atau oleh metode atau alat kuantitatif standar.
2. Dukungan untuk semua level manajerial, dari eksekutif puncak sampai manajer lini.
3. Dukungan untuk individu dan kelompok. Masalah yang kurang terstruktur sering memerlukan keterlibatan individu dari departemen dan tingkat organisasional yang berbeda atau bahkan dari organisasi lain. DSS mendukung im virtual melalui alat-alat Web kolaboratif.

4. Dukungan untuk keputusan independen dan atau sekuensial. Keputusan dapat dibuat satu kali, beberapa kali, atau berulang (dalam interval yang sama).
5. Dukungan disemua fase proses pengambilan keputusan : intelegensi, desain, pilihan, dan implementasi
6. Dukungan diberbagai proses dan gaya pengambilan keputusan.
7. Adaptivitas sepanjang waktu. Pengambil keputusan seharusnya reaktif, dapat menghadapi perubahan kondisi secara cepat, dan dapat mengadaptasi DSS untuk memenuhi perubahan tersebut. DSS bersifat fleksibel dan karena itu pengguna dapat menambahkan, menghapus, menggabungkan, mengubah, atau menyusun kembali elemen-elemen dasar, DSS juga fleksibel dalam hal dapat dimodifikasi untuk memecahkan masalah lain yang sejenis.
8. Peningkatan terhadap keefektifan pengambilan keputusan (akurasi, *timeliness*, kualitas).
9. Kontrol penuh oleh pengambil keputusan terhadap semua langkah proses pengambilan keputusan dalam memecahkan suatu masalah. DSS secara khusus menekankan untuk mendukung pengambil keputusan, bukannya menggantikan.

## II.2.2. Komponen Sistem Pendukung Keputusan

Sistem Pendukung Keputusan disusun dari beberapa subsistem yaitu :

1. Subsistem manajemen data

Basis data yang relevan dan dikelola menggunakan *software* yang disebut *database management system* (DBMS)

2. Subsistem manajemen model

Adalah paket *software* yang berisi model-model yang disebut dengan *modelbase management system* (MBMS)

3. Subsistem manajemen pengetahuan

Subsistem yang memberikan intelegensi dan mendukung subsistem yang lain.

4. Subsistem antarmuka pengguna

Pengguna berkomunikasi dan memerintah SPK melalui system ini.

5. Pengguna

Orang yang berhadapan dengan pengambil keputusan (Sri Hartati ; 2011 : 37).

### II.3. Metode *Simple Additive Weight* (SAW)

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.

$$r_{ij} = \begin{cases} \frac{x_{ij}}{\text{Max}_i x_{ij}} & \text{jika } j \text{ atribut keuntungan (benefit)} \\ \frac{\text{Min}_i x_{ij}}{x_{ij}} & \text{jika } j \text{ atribut biaya (cost)} \end{cases} \dots\dots\dots (1)$$

Keterangan :

1.  $R_{ij}$  = nilai rating kinerja normalisasi
2.  $X_{ij}$  = nilai atribut yang dimiliki dari setiap kriteria
3.  $\text{Max } x_{ij}$  = nilai terbesar dari setiap kriteria

4. Min  $x_{ij}$  = nilai terkecil dari setiap kriteria
5. Benefit = nilai terbesar adalah terbaik
6. Cost = nilai terkecil adalah terbaik

Dimana  $r_{ij}$  adalah rating kinerja ternormalisasi dari alternatif  $A_i$  pada atribut  $C_j$ ;  $i=1,2,\dots,m$  dan  $j=1,2,\dots,n$ . Nilai *preferensi* untuk setiap alternatif ( $V_i$ ) diberikan sebagai:

$$V_i = \sum_{j=1}^n w_j r_{ij} \dots\dots\dots (2)$$

Keterangan :

1.  $V_i$  = ranking untuk setiap alternatif
2.  $w_j$  = nilai bobot dari setiap kriteria
3.  $r_{ij}$  = nilai rating kinerja ternormalisasi.

Nilai  $V_i$  yang lebih besar mengindikasikan bahwa alternatif  $A_i$  lebih terpilih (Youllia Indrawaty ; 2011 : 34).

### II.3.1. Langkah *Simple Additive Weight* (SAW)

Langkah-langkah dari metode SAW adalah :

1. Menentukan kriteria-kriteria yang akan dijadikan acuan dalam pengambilan keputusan, yaitu C.
2. Menentukan rating kecocokan setiap alternatif pada setiap kriteria.
3. Membuat matriks keputusan berdasarkan kriteria (C), kemudian melakukan normalisasi matriks berdasarkan persamaan yang disesuaikan dengan jenis

atribut (atribut keuntungan ataupun atribut biaya) sehingga diperoleh matriks ternormalisasi R.

4. Hasil akhir diperoleh dari proses perankingan yaitu penjumlahan dari perkalian matriks ternormalisasi R dengan vector bobot sehingga diperoleh nilai terbesar yang dipilih sebagai alternatif terbaik (A) sebagai solusi (Youllia Indrawaty ; 2011 : 34).

### **II.3.2. Kelebihan Metode *Simple Additive Weighting* (SAW)**

Kelebihan dari model *Simple Additive Weighting* (SAW) dibandingkan dengan model pengambilan keputusan yang lain terletak pada kemampuannya untuk melakukan penilaian secara lebih tepat karena didasarkan pada nilai kriteria dan bobot preferensi yang sudah ditentukan, selain itu SAW juga dapat menyeleksi alternatif terbaik dari sejumlah alternatif yang ada karena adanya proses perankingan setelah menentukan nilai bobot untuk setiap atribut (Destriyana Darmastuti ; 2011 : 3).

### **II.4. Pengertian Visual Basic**

Visual basic dibuat oleh Microsoft, merupakan salah satu bahasa pemrograman berorientasi objek yang mudah dipelajari. Selain menawarkan kemudahan, Visual Basic juga cukup andal untuk digunakan dalam pembuatan berbagai aplikasi, terutama aplikasi *database*. Visual basic merupakan bahasa pemrograman *event drive*, di mana program aplikasi yang dapat berupa kejadian atau *event*, misalnya ketika *user* mengklik tombol atau menekan enter. Jika kita

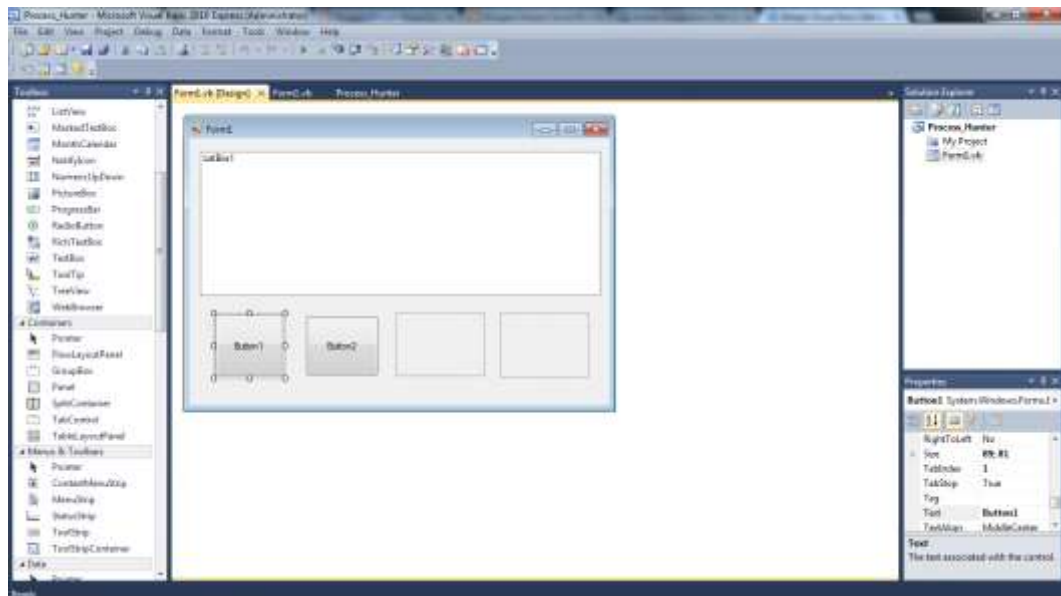
membuat aplikasi dengan Visual Basic maka kita akan mendapatkan *file* yang menyusun aplikasi tersebut, yaitu :

1. File Project (\*.vbp)

File ini merupakan kumpulan dari aplikasi yang kita buat. File project bisa berupa file \*.frm, \*.dsr atau file lainnya.

2. File Form (\*.frm)

File ini merupakan file yang berfungsi untuk menyimpan informasi tentang bentuk form maupun *interface* yang kita buat (Edy Winarno ; 2010 : 83).



**Gambar II.1. Tampilan VB.Net**  
(Sumber : Edy Winarno ; 2010 : 83)

## II.5. Pengertian SQL Server 2008

SQL Server 2008 adalah sebuah terobosan baru dari Microsoft dalam bidang *database*. SQL Server adalah DBMS (*Database Management System*) yang dibuat oleh Microsoft untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan Oracle. SQL Server

2008 dibuat pada saat kemajuan dalam bidang *hardware* sedemikian pesat. Oleh karena itu sudah dapat dipastikan bahwa SQL Server 2008 membawa beberapa terobosan dalam bidang pengolahan dan penyimpanan data.

Microsoft merilis SQL Server 2008 dalam beberapa versi yang disesuaikan dengan segment-segment pasar yang dituju. Versi-versi tersebut adalah sebagai berikut. Menurut cara pemrosesan data pada prosesor maka Microsoft mengelompokkan produk ini berdasarkan 2 jenis yaitu :

1. Versi 32-bit(x86), yang biasanya digunakan untuk komputer dengan single prosesor (Pentium 4) atau lebih tepatnya prosesor 32 bit dan sistem operasi Windows XP.
2. Versi 64-bit(x64), yang biasanya digunakan untuk komputer dengan lebih dari satu prosesor (Misalnya Core 2 Duo) dan system operasi 64 bit seperti Windows XP 64, Vista, dan Windows 7.

Sedangkan secara keseluruhan terdapat versi-versi seperti berikut

1. Versi Compact, ini adalah versi “Tipis” dari semua versi yang ada. Versi ini seperti versi desktop pada SQL Server 2000. Versi ini juga digunakan pada handled drvice seperti Pocket PC, PDA, SmartPhone, Tablet PC.
2. Versi Express, ini adalah versi “Ringan” dari semua versi yang ada(tetapi versi ini berbeda dengan versi compact) dan paling cocok untuk latihan para pengembang aplikasi. Versi ini memuat Express Manager standar, integrasi dengan CLR dan XML (Wenny Widya ; 2012 : 3)


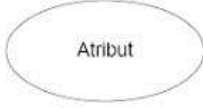



**Gambar II.2. Tampilan SQL Server**  
(Sumber : Wenny Widya ; 2012 : 3)

## **II.6. Entity Relationship Diagram (ERD)**

*Entity Relationship Diagram* atau ERD adalah alat pemodelan data utama dan akan membantu mengorganisasi data dalam suatu proyek ke dalam entitas-entitas dan menentukan hubungan antarentitas. Proses memungkinkan analisis menghasilkan struktur basisdata yang baik sehingga data dapat disimpan dan diambil secara efisien (Janner Simarmata ; 2010 : 67).

**Tabel II.1. Simbol ERD**

Notasi	Keterangan
	<b>Entitas</b> , adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
	<b>Relasi</b> , menunjukkan adanya hubungan di antara sejumlah entitas yang berbeda.
	<b>Atribut</b> , berfungsi mendeskripsikan karakter entitas (atribut yg berfungsi sebagai key diberi garis bawah)
	<b>Garis</b> , sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

(Sumber : Janner Simarmata ; 2010 : 67)

## II.7. Teknik Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi dari tabel rasional.

Teori normalisasi didasarkan pada konsep bentuk normal. Sebuah tabel relasional dikatakan berada pada bentuk normal tertentu jika tabel memenuhi himpunan batasan tertentu. Ada lima bentuk normal yang telah ditemukan.

**Adapun bentuk-bentuk normalisasi tersebut adalah :**

**1. Bentuk normal tahap pertama (1<sup>st</sup> Normal Form)**

Contoh yang kita gunakan di sini adalah sebuah perusahaan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri.

**2. Bentuk normal tahap kedua (2<sup>nd</sup> normal form)**

Definisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1NF, tetapi tidak pada 2NF. Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada bentuk normal kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada seluruh kolom yang membentuk kunci utama.

**3. Bentuk normal tahap ketiga (3<sup>rd</sup> normal form)**

Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional tergantung hanya pada kunci utama. Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya.

**4. *Boyce Code Normal Form* (BCNF)**

Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3NF sudah cukup

karena sangat jarang entitas yang berada pada 3NF bukan merupakan 4NF dan 5NF.

### **5. Bentuk Normal Tahap Keempat dan Kelima**

Sebuah tabel relasional berada pada bentuk normal keempat (4NF) jika dia dalam BCNF dan semua ketergantungan multivalued merupakan ketergantungan fungsional. Bentuk normal keempat (4NF) didasarkan pada konsep ketergantungan multivalued (MVD).

Sebuah tabel berada pada bentuk normal kelima (5NF) jika ia tidak dapat mempunyai dekomposisi lossless menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*) (Janner Simarmata ; 2010 : 76).

## **II.8. UML (*Unified Modeling Language*)**

Menurut Windu Gata (2013 : 4) 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


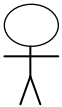
bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.



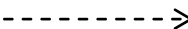
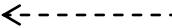
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasis UML adalah *use case* diagram, *activity* diagram, *sequence* diagram dan *class* diagram.

### II.8.1. Use case Diagram

*Use case* diagram merupakan pemodelan untuk kelakuan (*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, yaitu :

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

Gambar	Keterangan
	<p><i>Use case</i> menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama <i>use case</i>.</p>
	<p>Aktor adalah <i>abstraction</i> dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasi 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 <i>use case</i>, tetapi tidak memiliki control terhadap <i>use case</i>.</p>




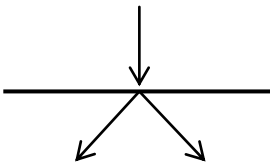
	Asosiasi antara aktor dan <i>use case</i> , digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan aliran data.
	Asosiasi antara aktor dan <i>use case</i> yang menggunakan panah terbuka untuk mengindikasikan bila aktor berinteraksi secara pasif dengan sistem.
	<i>Include</i> , merupakan di dalam <i>use case</i> lain ( <i>required</i> ) atau pemanggilan <i>use case</i> oleh <i>use case</i> lain, contohnya adalah pemanggilan sebuah fungsi program.
	<i>Extend</i> , merupakan perluasan dari <i>use case</i> lain jika kondisi atau syarat terpenuhi.

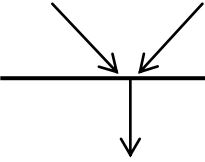
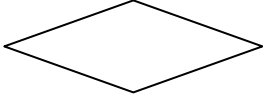
(Sumber : Windu Gata ; 2013 : 4)

## II.8.2. 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*, yaitu :

**Tabel II.3. Simbol *Activity Diagram***

Gambar	Keterangan
	<i>Start point</i> , diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	<i>End point</i> , akhir aktifitas.
	<i>Activites</i> , menggambarkan suatu proses/kegiatan bisnis.
	<i>Fork</i> (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.

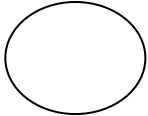
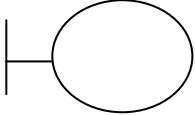

	<i>Join</i> (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	<i>Decision Points</i> , menggambarkan pilihan untuk pengambilan keputusan, <i>true</i> , <i>false</i> .
<div style="border: 2px solid black; padding: 2px; display: inline-block;">New Swimlane</div>	<i>Swimlane</i> , pembagian <i>activity diagram</i> untuk menunjukkan siapa melakukan apa.

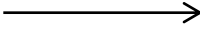
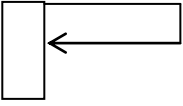

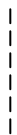
(Sumber : Windu Gata ; 2013 : 6)

### II.8.3. 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*, yaitu :

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

Gambar	Keterangan
	<i>Entity Class</i> , merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	<i>Boundary Class</i> , berisi kumpulan kelas yang menjadi <i>interface</i> atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan <i>form</i> cetak.
	<i>Control class</i> , suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.

	<i>Message</i> , simbol mengirim pesan antar <i>class</i> .
	<i>Recursive</i> , menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	<i>Activation</i> , <i>activation</i> mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	<i>Lifeline</i> , garis titik-titik yang terhubung dengan objek, sepanjang <i>lifeline</i> terdapat <i>activation</i> .

(Sumber : Windu Gata ; 2013 : 7)

#### II.8.4. 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 tanggung 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.

**Tabel II.5. Multiplicity Class Diagram**

<b>Multiplicity</b>	<b>Penjelasan</b>
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 : Windu Gata ; 2013 : 9)*