BAB II

LANDASAN TEORI

II.1. Game

Game merupakan kata dalam bahasa Inggris yang berarti permainan. Permainan adalah sesuatu yang dapat dimainkan dengan aturan tertentu sehingga ada yang menang dan ada yang kalah, biasanya dalam konteks tidak serius atau dengan tujuan refreshing. Dalam penggunaannya kata game sering digunakan untuk menyebutkan video game. Video game adalah game yang dimainkan dengan memanfaatkan media visual elektronik.

Sejarah video game tidak hanya mengenai orang-orang yang berperan di dalamnya, tetapi juga mengenai berbagai perusahaan game yang mempunyai kasus ironis. Atari adalah perusahaan Amerika dengan nama Jepang, dan perusahaan Jepang bernama SEGA didirikan oleh orang Amerika. Magnavox yang memulai peran perusahaan ini telah berumur 1 abad, dan Nintendo sebagai perusahaan yang mempopulerkan kembali video game juga sama tuanya, serta tidak ada yang pernah berpikir bahwa Sony yang merupakan perusahaan penemu banyak barang-barang elektronik mulai dari transistor radio hingga alat perekam video, akan membuat sebuah console yang menjadi produk dengan tingkat penjualan yang tinggi hingga saat ini. **(***Tri Daryanto, 2007***)**

Pada tahun 1952, seorang mahasiswa Universitas Cambridge bernama A.S. Gouglas membuat permainan OXO (tic-tac-toe) dalam versi grafik. Permainan ini ia kembangkan ketika hendak mendemonstrasikan tesisnya tentang

interaksi antara manusia dan komputer. Pada tahun 1958 William Higin Botham mendesain sebuah game dengan judul Tennis For Two yang dimainkan dalam oscilloscope, dan kemudian ada pula Steve Russel pada tahun 1961 dengan game berjudul Spacewar yang dibuat dalam komputer mainframe DEC PDP-1 saat mereka menjalani studi di MIT.

II.2. Game Merpati Shooter

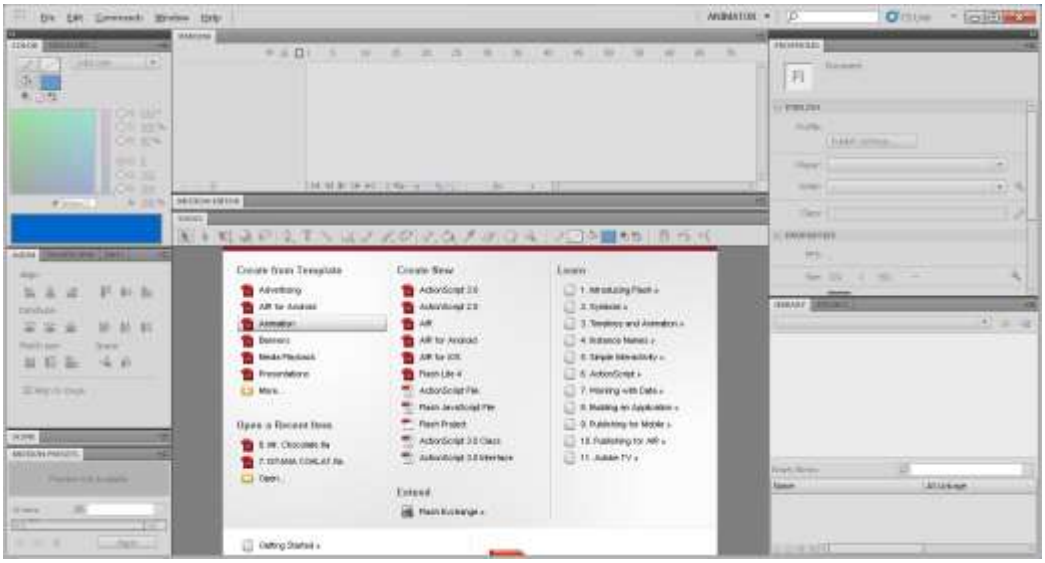
Game merpati shooter merupakan game yang berjenis FPS (*First Person Shooter*), game merpati shooter merupakan game dimana pemain harus menembak burung sebanyak mungkin hingga mencapai skor tertentu dan masuk ke level berikutnya, pada permainan ini terdapat 3 (tiga) buah level.

II.3. Adobe Flash

Adobe Flash CS5 adalah salah satu perangkat lunak komputer yang merupakan produk unggulan Adobe Systems. Adobe Flash CS5 digunakan untuk membuat gambar vector maupun animasi gambar tersebut. Berkas yang dihasilkan dari perangkat lunak ini mempunyai file *extension* .SWF dan dapat diputar di penjelajah web yang telah dipasangi Adobe Flash Player. Flash menggunakan bahasa pemrograman bernama ActionScript yang muncul pertama kalinya pada Flash 5.

Sebelum tahun 2005, Flash dirilis oleh Adobe. Flash 1.0 diluncurkan pada tahun 1996 setelah Adobe membeli program animasi vector bernama Future Splash. Versi terakhir yang diluncurkan di pasaran dengan menggunakan nama 'Adobe' adalah Adobe Flash 8. Pada tanggal 3 Desember 2005 Adobe Systems

mengakuisisi Adobe dan seluruh produknya, sehingga nama Adobe Flash berubah menjadi Adobe Flash, Dibawah ini Gambar II.1 tampilan Adobe Flash.



Gambar II.1 Tampilan Awal Adobe Flash

Adobe Flash merupakan sebuah program yang didesain khusus oleh Adobe dan program aplikasi standar authoring tool professional yang digunakan untuk membuat animasi dan bitmap yang sangat menarik untuk keperluan pembangunan situs web yang interaktif dan dinamis. Flash didesain dengan kemampuan untuk membuat animasi 2 dimensi yang handal dan ringan sehingga flash banyak digunakan untuk membangun dan memberikan efekanimasi pada website, CD Interaktif dan yang lainnya. Selain itu aplikasi ini juga dapat digunakan untuk membuat animasi logo, movie, game, pembuatan navigasi pada situs web, tombol animasi, banner, menu interaktif, interaktif form isian, e-card, screen saver dan pembuatan aplikasi-aplikasi web lainnya. Dalam Flash, terdapat teknik-teknik membuat animasi, fasilitas action script, filter, custom easing dan dapat memasukkan video lengkap dengan fasilitas playback FLV.

Keunggulan yang dimiliki oleh Adobe Flash ini adalah ia mampu diberikan sedikit code pemograman baik yang berjalan sendiri untuk mengatur animasi yang ada didalamnya atau digunakan untuk berkomunikasi dengan program lain seperti HTML, PHP, dan Database dengan pendekatan XML, dapat dikolaborasikan dengan web.

Ada 2 jenis animasi Macromedia Flash dapat dilakukan dengan dua cara yaitu *Frame by frame Animation* dan *Tweening animation,*yaitu:

1. *Frame By Frame Animation*

Bentuk dasar dari animasi adalah animasi frame per frame. Animasi frame per frame menuntut banyak gambar yang harus dibuat. Efek animasi diciptakan dengan mengganti gambar yang satu dengan gambar yang lain selama beberapa waktu. Semua gambar yang bergerak dihasilkan dari gambar yang berbeda-beda tiap framenya. Karena animasi frame per frame harus memiliki gambar yang unik tiap framenya maka animasi frame per frame sangat ideal untuk membuat animasi yang kompleks yang terdiri dari banyak perubahan seperti ekspresi wajah. Kelemahan dari animasi frame per frame adalah membutuhkan banyak waktu untuk membuat setiap gambar dan menghasilkan file yang besar ukurannya.

Didalam Flash, sebuah frame yang memiliki gambar yang unik dinamakan keyframe. Animasi frame per frame membutuhkan gambar yang unik setiap framenya, hal ini menyebabkan setiap framenya adalah keyframe.

2. *Tweening Animation*

Tweening animation sangat mengurangi waktu karena tidak perlu membuat animasi secara frame per frame. Sebaliknya hanya membuat frame awal dan frame akhir saja. Dua alasan utama mengapa tweening animation sangat baik

yaitu karena mengurangi pekerjaan mengambar dan meminimalkan ukuran file karena isi dari setiap frame tidak perlu disimpan.

Ada 2 jenis tweening animation yaitu Shape tween dan Motion tween, dimana masing-masing memiliki karakter yang unik.

1. Shape Tweening (Animasi Perubahan Bentuk)

Shape tweening berguna untuk mengubah bentuk. Flash hanya dapat mengubah bentuk, jadi tidak cocok untuk melakukan Shape tween untuk group, symbol, atau teks.

Shape Tween dapat dilakukan pada beberapa bentuk didalam sebuah layer, tetapi lebih baik untuk menempatkannya pada layer yang berbeda. Hal ini akan memudahkan jika ingin melakukan perubahan. Shape tweening juga memperbolehkan untuk mengubah warna.

2. Motion Tween

Motion tween tidak hanya berguna untuk menggerakkan groups, simbol, atau teks yang dapat diedit dari satu tempat ketempat lain. Motion tween dapat membantu untuk merubah ukuran, memutar, merubah warna dan transparan sisimbol. Motion tween hanya bisa digunakan pada satu objek pada satu layer. Jadi jika ingin mengerakkan banyak objek maka membutuhkan banyak layer.

II.3.1. Action Script

ActionScript adalah bahasa pemrograman yang dibuat berdasarkan ECMAScript, yang digunakan dalam pengembangan situs web dan perangkat lunak menggunakan platform Adobe Flash Player. ActionScript juga dipakai pada beberapa aplikasi basis data, seperti Alpha Five. Bahasa ini awalnya

dikembangkan oleh Macromedia, tapi kini sudah dimiliki dan dilanjutkan perkembangannya oleh Adobe, yang membeli Macromedia pada tahun 2005.

Perintah actionscript menyerupai sintaks penulisan bahasa C, penulisan ini digunakan untuk memudahkan pengembangan memberi perintah, contoh perintah action script seperti ini.

```
if (ar == 9)
{
   gotoAndStop(17);
} // end if
this.ss._visible = false;
this.salah._visible = false;
this.krt.onRelease = function ()
{
   function time()
   {
      gotoAndPlay(n);
      clearInterval(interval);
   } // End of the function
   if (this.hitTest(this._parent.tmp))
   {
      ++ar;
      n = arr1[ar];
      interval = setInterval(time, 1000);
   } // end if
```

```
        };
        stop ();
        sound = new Sound();
        sound.attachSound("yangmanaya");
    sound.start();
```

Perintah diatas merupakan salah satu jenis perintah actionscript yang digunakan untuk mengeluarkan suara

II.3.2. Komponen ActionScript

ActionScript seperti halnya bahasa pemrograman yang lain memiliki beberapa komponen penyusun. Ada beberapa komponen actionscript diantaranya adalah

1. **Komentar**

   Komentar merupakan bagian program yang tidak akan diproses atau dijalankan oleh compiler. Penulisan komentar selalu didahului oleh tanda 2 buah garis miring (//).

   Contoh: // ini adalah sebuah komentar

2. **Identifier**

   Identifier atau pengenal pada ActionScript bersifat case-sensitive yang berarti membedakan penggunaan huruf besar dan kecil. Selain menggunakan huruf, identifier juga dapat menggunakan angka atau underscore (_).

3. **Variabel dan Konstanta**

   Variabel merupakan nama untuk sebuah lokasi penyimpanan. Variabel harus dideklarasikan dengan menyebutkan nama dan tipe data dari informasi yang

akan disimpan. Sedangkan konstanta merupakan identifier yang serupa dengan variabel, namun digunakan untuk menyimpan nilai yang tidak dapat berubah.

Contoh: var timing:Boolean = false;

4. Tipe Data

Jenis-jenis tipe data pada ActionScript antara lain sebagai berikut:

a. Integer: berisi data semua bilangan bulat.
b. Array: disebut juga data bertingkat atau data yang mengandung beberapa data lagi di dalamnya dan diindeks berdasarkan data numerik atau string.
c. String: digunakan untuk menampung angka atau huruf.
d. Boolean: tipe data yang hanya terdiri dari dua kemungkinan nilai, yaitu true (benar) atau false (salah).
e. MovieClip: merupakan tipe data yang digunakan untuk mengontrol simbol movie clip dengan menggunakan method dari MovieClip Class.
f. Null: tipe data yang tidak menyimpan suatu data apa pun atau kosong (null).
g. Number: dapat mewakili integer maupun bilangan floating point.
h. Object: tipe data yang digunakan untuk memberi definisi kepada suatu Objek Class.
i. Undefined
j. Void

(Sumber: http://www.scribd.com/doc/88799111/2/MENGENAL-TIPE-DATA):

II.4. Multimedia

Multimedia berasal dari kata “multi” dan “media”. Multi berarti

banyak dan media adalah merupakan sarana untuk penerapan. Multimedia dapat diartikan sebagai penggunaan beberapa media yang berbeda untuk menggabungkan dan menyampaikan informasi dalam bentuk teks, gambar/grafik, animasi, *audio* dan video (dapat disebut juga gabungan dari berbagai media yang terintegrasi). Beberapa definisi menurut beberapa ahli :

1. Kombinasi dari komputer dan video .
2. Kombinasi dari tiga elemen: suara, gambar, dan teks.
3. Kombinasi dari paling sedikit dua media *input* atau *output*. Media ini dapat berupa *audio* (suara, musik), animasi, video, teks, grafik dan gambar.
4. Alat yang dapat menciptakan presentasi yang dinamis dan interaktif yang mengkombinasikan teks, grafik, animasi, *audio* dan video.
5. Multimedia dalam konteks komputer menurut Hofstetter (2001) adalah pemanfaatan komputer untuk menggabungkan teks, grafik, *audio*, video, dengan menggunakan *tool* yang memungkinkan pemakai berinteraksi, berkreasi dan berkomunikasi. *(Sumber : Iwan Binanto, Multimedia Digital - Dasar Teori Dan Pengembangannya, 2010)*

Adapun beberapa karakteristik multimedia adalah sebagai berikut :

1. Bersifat fleksibel (memberikan keleluasaan bagi *user* untuk memilih materi dan menggunakannya)
2. Bersifat *self-pacing* (memberikan kesempatan user untuk belajar sesuai dengan kecepatannya dalam memahami materi)
3. Bersifat *content-rich* (memeberikan informasi yang kaya baik dari isi maupun medianya).
4. Bersifat interaktif (memberikan kesempatan bagi *user* untuk melakukan

respon dan mencoba simulasi).

II.4.1. Suara

Suara adalah fenomena fisik yang dihasilkan oleh getaran benda, getaran suatu benda yang berupa sinyal analog dengan amplitudo yang berubah secara kontinyu terhadap waktu. Konsep dasar suara dihasilkan oleh getaran suatu benda. Selama bergetar perbedaan tekanan terjadi di udara sekitarnya ini yang disebut dengan Gelombang. Gelombang mempunyai pola sama yang berulang pada interval tertentu, yang disebut sebagai "Periode".

Suara yang berada pada batas pendengaran manusia sebagai "a*udio*", dan gelombangnya sebagai "a*ccoustic signals*". Suara di luar batas pendengaran manusia dapat dikatakan sebagai "n*oise*" (getaran yang tidak teratur dan tidak berurutan dalam berbagai frekuensi, tidak dapat di dengar manusia).

Format audio yang ada saat ini antara lain : AAC (*Advanced Audio Coding)* [.m4a], *WAVEFORM AUDIO* [.WAV], *Audio Interchange File Format* [.AIF], *Audio CD* [.cda], *Mpeg Audio Layer 3* [.mp3], MIDI (*Music Instrument Digital Interface*) dan sebagainya.

Berdasarkan frekuensi, suara dibagi menjadi 6 bagian yaitu :

1. Infrasound 0Hz – 20 Hz
2. Pendengaran manusia 20 Hz – 20 KHz
3. Ultrasound 20 KHz – 1 GHz
4. Hypersound 1 GHz – 10 THz
5. Manusia membuat suara dengan frekuensi : 50 Hz – 10 KHz.

6. Sinyal suara musik memiliki frekuensi : 20 Hz – 20 KHz.

Sistem multimedia menggunakan suara yang berada dalam range pendengaran manusia. Sebelum kerja animasi sebenar bermula, direkamnya satu runut bunyi permulaan atau "*scratch track*" supaya animasi akan digerakkan dengan runut bunyi dengan lebih jitu. Oleh sebab kaedah penerbitan animasi tradisional yang perlahan dan teratur adalah lebih mudah untuk menggerakkan animasi dengan runut bunyi yang sedia ada daripada menggerakkan runut bunyi dengan hasil animasi yang ada. Runut bunyi animasi yang lengkap terdiri daripada muzik, efek bunyi, dan dialog yang dipersembahkan oleh pelakon suara. Bagaimanapun runut bunyi permulaan yang digunakan ketika animasi biasanya hanya mengandungi suara-suara pelakon, lagu-lagu vokal yang perlu dinyanyikan oleh pelakon, dan skor muzik sementara; skor muzik yang tetap serta efek bunyi diterapkan dalam pasca penerbitan.

Bagi kebanyakan kartun animasi bersuara yang dibikin sebelum tahun 1930, bunyinya *dipascagerakkan*, yaitu runut bunyi dirakam selepas animasi gambar disiapkan dengan menonton film sambil mempersembahkan dialog, muzik, dan efek bunyi yang dikehendaki. Setengah studio masih mempascagerakkan kartun-kartun terbitannya sepanjang 1930-an, terutamanya Fleischer Studios, yang mana "gumaman *ad-lib*" diterapkan dalam kartun-kartun seperti *Popeye the Sailor* dan *Betty Boop*.

II.4.2. Gambar

Gambar merupakan suatu representasi *spatial* dari suatu obyek, dalam pandangan 2D atau 3D. Gambar digital merupakan suatu fungsi

dengan nilai-nilai yang berupa intensitas cahaya pada tiap–tiap titik pada bidang yang telah diquantisasikan (diambil sampelnya pada interval diskrit).

Titik dimana suatu gambar di-*sampling* disebut *picture element* (*pixel*). Nilai intensitas warna pada suatu *pixel* disebut *gray scale level*.

1 bit binary-valued image (0 - 1)

8 bits gray level (0 - 255)

16 bits high color (2^{16})

24 bits True Color (2^{24})

32 bits true color (2^{32})

Format file gambar yang ada saat ini antara lain : *Bitmap* (BMP), *Joint Photographic Expert Group* (JPEG/JPG), *Graphics Interchange Format* (GIF), *Portable Network Graphics* (.PNG), *Tagged Image File Format* (TIFF), *Icon* (ICO), *Enchanced Windows Metafile* (EMF), PCX, ANI (*Animation*), *Cursor* (CUR), WBMP (WAP BMP), *Adobe Photoshop Document* (PSD), dan *Corel Draw* (CDR).

II.5. Unified Modelling Language

Unified Modeling Language (UML) merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek (Whitten L. Jeffery et al, 2004). Sementara menurut Henderi (2007) UML adalah sebuah bahasa pemodelan yang telah menjadi standar dalam industri software untuk visualisasi, merancang, dan mendokumentasikan sistem perangkat lunak. Bahasa Pemodelan UML lebih cocok untuk pembuatan perangkat lunak dalam bahasa pemrograman berorientasi

objek (C++, Java, VB.NET), namun demikian tetap dapat digunakan pada bahasa pemrograman prosedural (Djon Irwanto, Perancangan Object Oriented Software dengan UML, 2007).

UML merupakan pemodelan objek yang fokus pada pendefinisian struktur statis dan model sistem informasi yang dinamis daripada mendefinisikan data dan model proses yang tujuannya adalah pengembangan tradisional. UML menawarkan diagram yang dikelompokan menjadi lima perspektif berbeda untuk memodelkan suatu sistem. Adapun daigram yang terdapat dalam uml adalah sebagai berikut (Djon Irwanto, Perancangan Object Oriented Software dengan UML, 2007):

1. **Model *Use Case* Diagram**

Use Case Diagram secara grafis menggambarkan interaksi antara sistem, sistem eksternal, dan pengguna. Dengan kata lain Use Case diagram secara grafis mendeskripsikan siapa yang akan menggunakan sistem dan dalam cara apa pengguna (user) mengharapkan interaksi dengan sistem itu. Use Case secara naratif digunakan ntuk secara tekstual menggambarkan sekuensi langkah-langkah dari setiap interaksi.

2. ***Class* Diagram**

Menggambarkan struktur object sistem. Diagram ini menunjukan class object yang menyusun sistem dan juga hubungan antara class object tersebut.

3. ***Sequence* Diagram**

Secara grafis menggambarkan bagaimana objek berinteraksi dengan satu sama lain melalui pesan pada sekuensi sebuah use case atau operasi. Diagram ini

mengilustrasikan bagaimana pesan terkirim dan diterima di antara objek dan dalam sekuensi atau timing apa.

4. *Diagram aktivitas/Activity Diagram*

Secara grafis digunakan untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun use case. Activity diagram dapat juga digunakan untuk memodelkan action yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari action tersebut.

UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan bahasa pemodelan visual yang memungkinkan pengembang sistem membuat *blue print* atas visinya dalam bentuk yang baku. UML berfungsi sebagai jembatan dalam mengkomunikasikan beberapa aspek dalam sistem melalui sejumlah elemen grafis yang bisa dikombinasikan menjadi diagram. UML mempunyai banyak diagram yang dapat mengakomodasi berbagai sudut pandang dari suatu perangkat lunak yang akan dibangun. Diagram-diagram tersebut digunakan untuk:

1. Mengkomunikasikan ide
2. Melahirkan ide-ide baru dan peluang-peluang baru
3. Menguji ide dan membuat prediksi
4. Memahami struktur dan relasi-relasinya

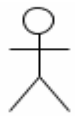
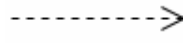
(Djon Irwanto, Perancangan Object Oriented Software dengan UML, 2007)

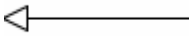
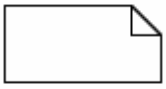
Berdasarkan uraian di atas maka penulis membuat sebuah alur sistem yang di tampilkan dalam bentuk *Use Case diagram*, *Activity diagram*, dan *Class diagram* dalam model *Unified Modelling Language* (UML).

II.5.1. *Use Case diagram*

Use Case diagram adalah model fungsional sebuah sistem yang menggunakan *actor* dan *use case*. *Use Case* adalah layanan (*services)* atau fungsi-fungsi yang disediakan oleh sistem untuk pengguna-penggunanya (Djon Irwanto, Perancangan Object Oriented Software dengan UML, 2007). *Use Case* adalah suatu pola atau gambaran yang menunjukan kelakukan atau kebiasaan sistem. Setiap *Use Case* adalah suatu urut-urutan (*sequence*) transaksi yang saling berhubungan dan dilakukan oleh sebuah *actor* dan sistem dalam bentuk sebuah dialog (Henderi, 2007). *Use Case diagram* dibuat untuk memvisualisasikan/ menggambarkan hubungan antara *Actor* dan *Use Case*. *Use Case diagram* mempresentasikan kegunaan atau fungsi-fungsi sistem dari perspektif pengguna. Simbol-simbol yang digunakan dalam *Use Case diagram* adalah sebagai berikut :

Tabel 2.2 Daftar simbol dalam Use Case diagram

GAMBAR	NAMA	KETERANGAN
	*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.

	*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
	Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.
	*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
	*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
	*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Sumber : Djon Irwanto, Perancangan Object Oriented Software dengan UML, 2007

II.5.2. *Activity diagram*

Activity diagram adalah diagram yang menggambarkan sifat dinamis secara alamiah sebuah sistem dalam bentuk model aliran dan kontrol dari aktivitas ke aktivitas lainnya (*Sumber : Djon Irwanto, Perancangan Object Oriented Software dengan UML, 2007*). Karena sebuah *activity diagram* adalah bentuk khusus dari *statechart diagram*, maka *activity diagram* kadangkala digunakan untuk memodelkan suatu kebiasaan sesuai dengan ketentuan/kaedah bisnis.Simbol-simbol yang digunakan dalam *Activity diagram* adalah sebagai berikut :

Tabel 2.3 Daftar simbol dalam Activity diagram

GAMBAR	NAMA	KETERANGAN
	*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	*Action*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi.
	*Initial Node*	Bagaimana objek dibentuk atau diawali.
	*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan.
	*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran.

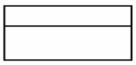
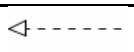
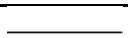
Sumber : Djon Irwanto, Perancangan Object Oriented Software dengan UML, 2007

II.5.3. *Class diagram*

Class adalah kumpulan objek-objek dengan dan yang mempunyai struktur umum, *behavior* umum, relasi umum, dan *semantic*/kata yang umum. *Class-class* ditentukan/ditemukan dengan cara memeriksa objek-objek dalam *sequence diagram* dan *collaboration diagram*. Sebuah *class* digambarkan seperti sebuah bujur sangkar dengan tiga bagian ruangan.

Class diagram adalah diagram yang menunjukan *class-class* yang ada dari sebuah sistem dan hubungannya secara logika. *Class diagram* menggambarkan struktur statis dari sebuah sistem. Karena itu *class diagram* merupakan tulang punggung atau kekuatan dasar dari hampir setiap metode berorientasi objek termasuk UML (*Djon Irwanto, Perancangan Object Oriented Software dengan UML, 2007*).

Tabel 2.4 Daftar simbol dalam Class diagram

GAMBAR	NAMA	KETERANGAN
	Generalization	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*N-Ary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

Sumber : Djon Irwanto, Perancangan Object Oriented Software dengan UML, 2007