BAB 2

TINJAUAN PUSTAKA

2.1 Tinjauan Perusahaan

Tinjauan perusahaan yang dijelaskan meliputi deskripsi mengenai Distro Heaven Skateboards, logo Distro Heaven Skateboards, struktur organisasi, serta deskripsi tugas.

2.1.1 Distro Heaven Skateboards

Pendirian Heaven Skateboards bersamaan dengan pembangunan Skate Park pada bulan April 2006 yang berlokasi di Jalan Trunojoyo No. 15 Bandung. Distro ini menyediakan alat-alat *skateboard* seperti papan *skateboard*, *skate shoes*, *t'shirt*, jaket, topi, kacamata, dll.

Produk-produk di Heaven diimpor dari luar negeri, karena kayu yang digunakan untuk membuat papan *skate* adalah *maple* Canada yang memang jelas tidakada di Indonesia. Meskipun Heaven memperoduksi papan sendiri dengan brand Heaven Skateboard, proses pembuatannya pun dilakukan di Amerika.

Papan *skateboard* dijual terpisah dengan kelengkapan lainnya seperti *wheel* (roda), *truck*, dan *baring*. Untuk papan Amerika, harganya mulai Rp 500.000,- sampai dengan Rp 600.000,- sedangkan untuk produk asli Heaven Rp 400.000,-.

2.1.2 Logo Distro Heaven Skateboards

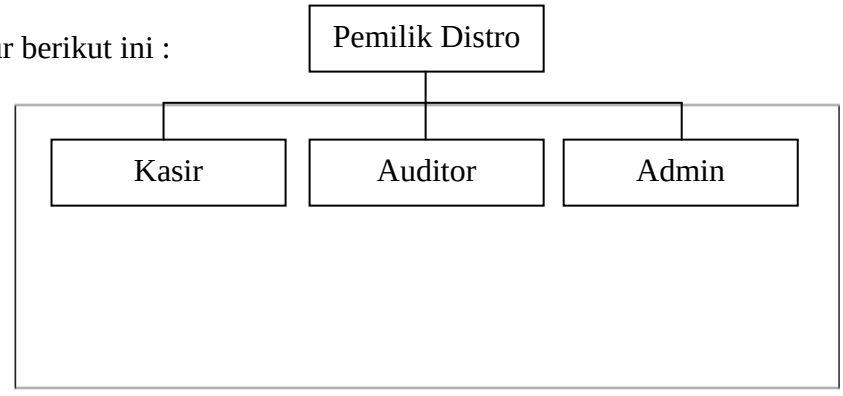
Di bawah ini merupakan logo dari Distro Heaven Skateboards yang dapat dilihat pada gambar 2.1 di halaman berikutnya.

Gambar 2. 1 Logo Heaven Skateboards

2.1.3 Struktur Organisasi Perusahaan

Struktur organisasi merupakan susunan yang terdiri dari fungsi-fungsi serta hubungan mengenai seluruh pernyataan setiap kegiatan untuk mencapai suatu tujuan. Secara fisik struktur organisasi dapat dinyatakan dalam bentuk bagan yang memperlihatkan hubungan unit-unit organisasi dan garis-garis wewenang yang ada.

Beberapa keuntungan yang dapat diperoleh dari penggunaan bagan organisasi yaitu agar dapat memperlihatkan karakteristik utama dari suatu perusahaan tersebut mengenai gambaran pekerjaan dan hubungan-hubungan yang ada di dalam perusahaan serta digunakan untuk merumuskan rencan kerja yang ideal sebagai pedoman untuk dapat mengetahui siapa bawahan dan atasannya. Struktur keorganisasian pada Distro Heaven Skateboards dapat dilihat pada bagan struktur berikut ini :



Gambar 2. 2 Struktur Organisasi

2.1.4 Deskripsi Tugas

Deskripsi tugas digunakan untuk mengetahui tugas, wewenang, serta tanggung jawab dari masing-masing bagian. Deskripsi tugas yang ada di Distro Heaven Skateboards adalah sebagai berikut :

1. Pemilik Distro
    a. Sebagai pengambil keputusan.
    b. Sebagai koordinator semua kegiatan yang telah dilaksanakan.
    c. Mengevaluasi semua kegiatan yang telah dilaksanakan.
    d. Menerapkan dan mengesahkan kebijakan yang menyangkut eksistensi toko.
2. Kasir
    a. Melayani pelanggan secara langsung.
    b. Menghitung hasil penjualan dan bertanggung jawab atas uang yang masuk.
    c. Melaporkan hasil penjualan secara harian, mingguan, bulanan, dan periodik.
    d. Melaporkan ke bagian auditor jika stok barang di cabang kurang atau habis.
3. Auditor
    a. Mengecek stok barang di tiap cabang.
    b. Mengirim stok barak ke tiap cabang.
    c. Mengecek laporan penjualan dari tiap cabang.
4. Admin
    a. Mengelola situs Heaven Skateboards.

b. Melayani pelanggan secara *online.*

2.2 Landasan Teori

Landasan teori merupakan ulasan atau penjabaran kembali teori-teori yang ada yang diambil dari berbagai sumber baik buku-buku, pustaka, maupun internet. Pengambilan teori-teori dari para pakar dibidangnya harus berkaitan dengan topik penelitian. Selain itu, pengambilan teori juga ditekankan pada sumber-sumber yang baru.

Landasan teori yang dijelaskan pada penelitian ini meliputi konsep dasar aksesories, konsep UML, *image processing*, *augmented reality*, cara kerja algoritma Haar Cascade Classifier, *library marilena*, starUML, adobe flash, *action script*, adobe photoshop cs, kuesioner dan skala.

2.2.1 Konsep Dasar Aksesories

Aksesori (atau aksesoris) adalah benda-benda yang dikenakan seseorang untuk mendukung atau menjadi pengganti pakaian. Bentuk aksesori bermacam-macam dan banyak di antaranya terkait dengan peran gender pemakainya.

Jenis aksesori bermacam-macam, seperti perhiasan (anting-anting atau giwang, kalung, gelang, bros), selendang, sabuk, suspender, dasi, syal, sarung tangan, sapu tangan, tas, topi, arloji, kacamata, dan pin. Busana tradisional memiliki aksesori khas yang biasanya dikenakan sebagai perlambang tertentu, seperti destar, sindur, tusuk konde, kembang goyang, dan keris. Di halaman berikutnya merupakan gambar dari macam-macam aksesories.



Gambar 2. 3 Aksesories

2.2.2 Konsep UML *(Unified Modeling Language)*

UML (*Unified Modeling Language*) merupakan pengganti dari metode analisis berorientasi objek dan desain berorientasi objek (OOA&D) yang dimunculkan sekitar akhir tahun 80-an dan awal tahun 90-an. UML merupakan gabungan dari metode Booch, Rumbaugh (OMT) dan Jacobson. Tetapi UML ini akan mencakup lebih luas daripada OOA&D. Pada pertengahan pengembangan UML dilakukan standarisasi proses dengan OMG (*Object Management Group*) dengan harapan UML akan menjadi bahasa standar pemodelan pada masa yang akan datang.

UML disebut sebagai bahasa pemodelan bukan metode. Kebanyakan metode terdiri paling sedikit prinsip, bahasa pemodelan dan proses. Bahasa pemodelan (sebagian besar grafik) merupakan notasi dari metode yang digunakan untuk mendesain secara cepat.

Bahasa pemodelan merupakan bagian terpenting dari metode. Ini merupakan bagian kunci tertentu untuk komunikasi. Jika anda ingin berdiskusi tentang desain dengan seseorang, maka Anda hanya membutuhkan bahasa pemodelan bukan proses yang digunakan untuk mendapatkan desain. UML merupakan bahasa standar untuk penulisan *blueprint software* yang digunakan untuk visualisasi, spesifikasi, pembentukan dan pendokumentasian alat-alat dari sistem perangkat lunak [3].

2.2.2.1 Sejarah UML

UML dimulai secara resmi pada oktober 1994, ketika Rumbaugh bergabung dengan Booch pada Relational Software Corporation. Proyek ini memfokuskan pada penyatuan metode Booch dan OMT. UML versi 0.8 merupakan metode penyatuan yang dirilis pada bulan Oktober 1995. Dalam waktu yang sama, Jacobson bergabung dengan Relational dan cakupan dari UML semakin luas sampai diluar perusahaan OOSE. Dokumentasi UML versi 0.9 akhirnya dirilis pada bulan Juni 1996. Meskipun pada tahun 1996 ini melihat dan menerima *feedback* dari komunitas *Software Engineering*.

Dalam waktu tersebut, menjadi lebih jelas bahwa beberapa organisasi perangkat lunak melihat UML sebagai strategi dari bisnisnya, kemudian dibangunlah UML Consortium dengan beberapa organisasi yang akan menyumbangkan sumber dayanya untuk bekerja, mengembangkan, dan melengkapi UML. Di sini beberapa *partner* yang berkontribusi pada UML 1.0, diantaranya Digital Equipment Corporation, Hewlett-Packard, I-Logix, Intellicorp, IBM, ICON Computing, MCI Systemhouse, Microsoft, Oracle,

Relational, Texas Instruments dan Unisys. Dari kolaborasi ini dihasilkan UML 1.0 yang merupakan bahasa pemodelan yang ditetapkan secara baik, *expressive*, kuat, dan cocok untuk lingkungan masalah yang luas. UML 1.0 ditawarkan menjadi standarisasi dari Object Management Group (OMG) dan pada Januari 1997 dijadikan sebagai standar bahasa pemodelan.

Antara Januari–Juli 1997 gabungan group tersebut memperluas kontribusinya sebagai hasil respon dari OMG dengan memasukkan Adersen Consulting, Ericsson, Object Time Limeted, Platinum Technology, Ptech, Reich Technologies, Softeam, Sterling Software dan Taskon. Revisi dari versi UML (versi 1.1) ditawarkan kepada OMG sebagai standarisasi pada bulan Juli 1997. Dan pada bulan September 1997,versi ini dierima oleh OMG *Analysis* dan *Design Task Force* (ADTF) dan OMG *Architecture Board* dan Akhirnya pada Juli 1997 UML versi 1.1 menjadi standarisasi.

Pemeliharaan UML terus dipegang oleh OMG *Revision Task Force* (RTF) yang dipimpin oleh Cris Kobryn. RTP merilis editorial dari UML 1.2 pada Juni 1998 dan pada tahun 1998 RTF juga merilis UML 1.3 disertai dengan *user guide* dan memberikan *technical cleanup* [3].

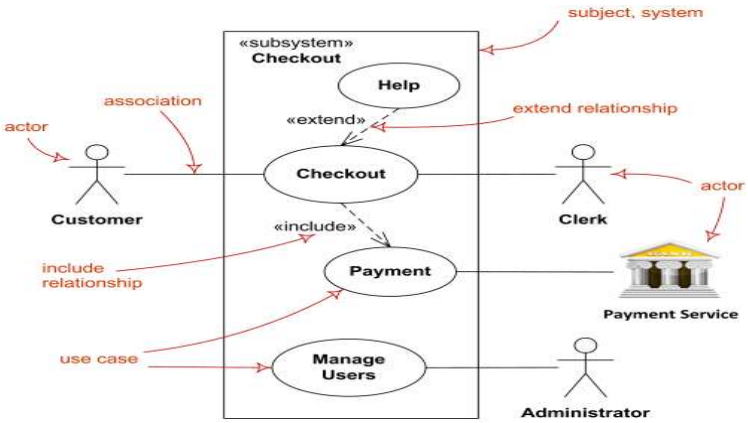
2.2.2.2 Tipe-Tipe Diagram UML

UML memiliki tipe-tipe diagram, diantaranya yaitu sebagai berikut :

a. *Use Case Diagram*

   Use Case Diagram digunakan untuk menggambarkan konteks sistem yang akan dibangun dan fungsionalitas yang disediakan oleh sistem itu. Use

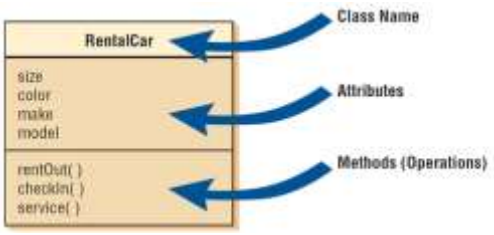
Case Diagram juga menggambarkan siapa (atau apa) berinteraksi dengan sistem. Di dalam sistem hanya terdapat satu *user*.



Gambar 2. 4 *Use Case Diagram*

b. *Class Diagram*

Class diagram digunakan untuk menunjukan hubungan dan eksistensi sebuah class dari sudut pandang logis dalam sistem. Sebuah Class Diagram merepresentasikan struktur class yang membentuk arsitektur dan perilaku sistem dalam proses desain dan analisis. *Class diagram* memberikan gambaran secara menyeluruh tentang object dan class serta relasinya pada arsitektur sistem yang dibangun. *Class diagram* menyediakan fasilitas berbagi model sehingga model yang telah dibangun bisa digunakan kembali (reusable) untuk pemodelan yang lebih rinci pada perilaku dinamis. Notasi pada *class diagram* bisa digunakan untuk menjelaskan *Object Oriented Programming*, sehingga semua konsep *Object Oriented Programming* bisa di jelaskan menggunakan notasi *Class diagram*.



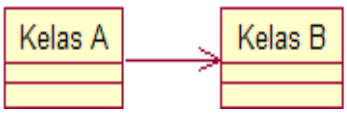
Gambar 2. 5 *Class Diagram*

1. Nama Kelas/*Stereotype*.
2. Atribut, yaitu variabel-variabel yang dimiliki suatu kelas.
3. Metode, yaitu fungsi-fungsi yang dimiliki oleh suatu kelas.

Class Diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. *Class Diagram* mempunyai 3 macam hubungan, sebagai berikut :

1. *Association*

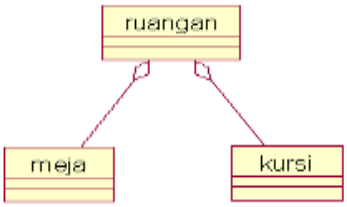
   Assosiation adalah hubungan statis antar kelas. Umumnya menggambarkan kelas yang memiliki atribut berupa kelas lain, atau kelas yang harus mengetahui ekstensi kelas lain. Dalam notasi UML kita mengenal asosiasi 2 arah (bidirectional) dan 1 arah (undirectional).



Gambar 2. 6 *Association*

2. *Aggregation*

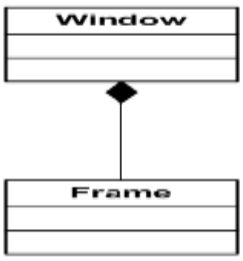
   Aggregation adalah *association* dimana salah satu kelasnya merupakan bagian dari suatu kumpulan. *Aggregation* memiliki titik pusat yang mencakup keseluruhan bagian.



Gambar 2. 7 *Aggregation*

3. *Composition*

   Composition adalah *aggregation* dengan ikatan yang lebih kuat. Di dalam *composite aggregation*, siklus hidup *part class* sangat bergantung pada *whole class* sehingga bila objek *instance* dari *whole class* dihapus maka objek *instance* dari *part class* juga akan terhapus.



Gambar 2. 8 *Composition*

4. *Depedency*

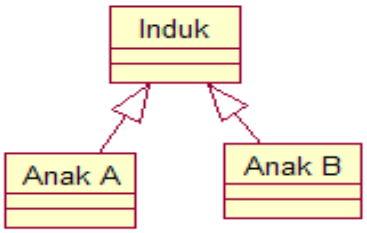
   Depedency adalah hubungan antar-class di mana sebuah class memiliki ketergantungan pada class lainnya tetapi tidak sebaliknya.



Gambar 2. 9 *Depedency*

5. *Generalization*

   Generalization diperlukan untuk memperlihatkan hubungan pewarisan (*inheritance*) antar unsur dalam diagram kelas. Pewarisan memungkinkan suatu kelas mewarisi semua atribut, operasi ,relasi, dari kelas yang berada dalam hirarki pewarisannya.



Gambar 2. 10 *Generalization*

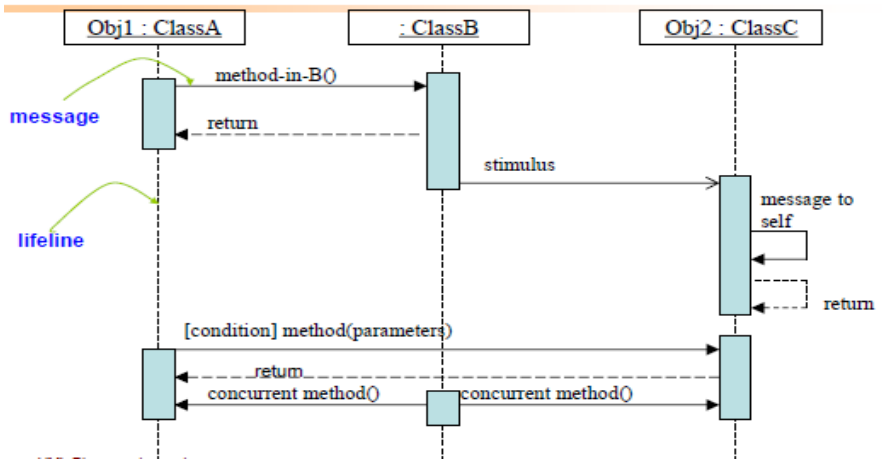
c. *Statechart Diagram*

   Statechart Diagram menggambarkan semua *state* yang dimiliki oleh suatu objek dari suatu class dan keadaan yang menyebabkan state berubah.

d. *Sequence Diagram*

   Sequence diagram merupakan suatu diagram yang menggambarkan interaksi antar objek dan mengindikasikan komunikasi diantara objek-objek tersebut. Diagram ini juga menunjukkan serangkaian pesan yang dipertukarkan oleh objek-objek yang melakukan suatu tugas atau aksi tertentu. Objek-objek tersebut kemudian diurutkan dari kiri ke kanan, aktor

yang menginisialisasi interaksi biasanya disimpan di paling kiri dari diagram.



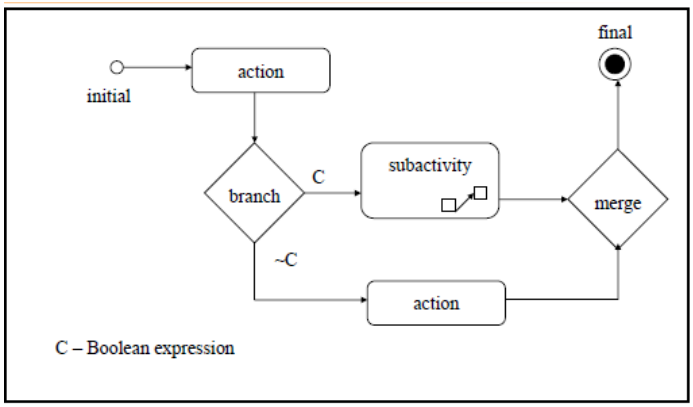
Gambar 2. 11 *Sequence Diagram*

e. *Collaboration Diagram*

Collaboration Diagram menggambarkan kolaborasi dinamis seperti *sequence diagram*. Dalam menunjukan pertukaran pesan, collaboration diagram menggambarkan objek dan hubungannya.

f. *Activity Diagram*

Activity diagram merupakan cara menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses yang dipakai pada *business modeling* untuk memperlihatkan urutan aktifitas proses bisnis. Struktur diagram ini mirip *flowchart* atau *Data Flow Diagram* (DFD) pada perancangan terstruktur. *Activity diagram* sangat bermanfaat apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan.



Gambar 2. 12 *Activity Diagram*

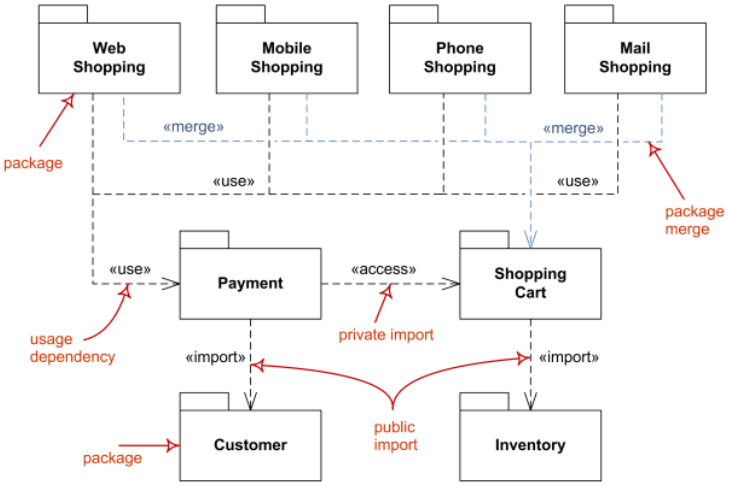
g. *Component Diagram*

Menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan diantaranya. Komponen dapat juga berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lainnya [3].

h. *Package Diagram*

Package diagram digunakan untuk menjelaskan hubungan modularitas, enkapsulasi, dan abstraksi pada pengembangan sistem yang kompleks. *Package diagram* juga digunakan untuk menjelaskan hubungan *dependency* dan *visibility* antar *class* yang memiliki asosiai secara modular dengan *class* lain dalam sistem. *Package diagram* memungkinkan perancangan sistem bisa di mengerti dengan tingkat pemahaman yang lebih tinggi pada kolaborasi antar elemen model. Dengan menggunakan *package diagram* perancangan *class diagram* bisa di modularisasi secara lebih terstruktur. *Package diagram* sangat dibutuhkan ketika *class diagram* yang dibangun sudah semakin kompleks, sehingga penggambaran

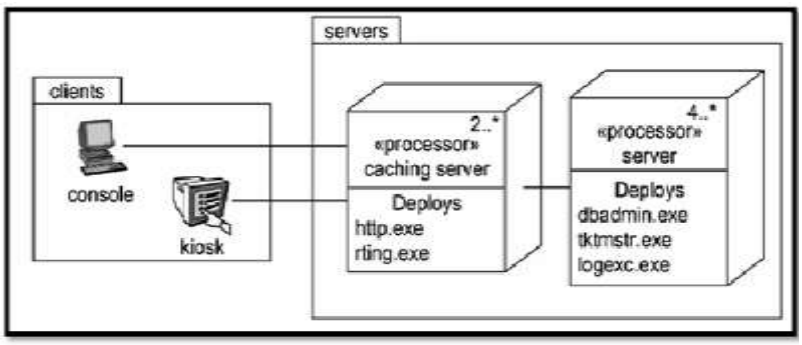
class diagram bisa dibangun secara terpisah dengan menggunakan *Package diagram* untuk mewakili hubungan yang antar class pada *package* yang berbeda.



Gambar 2. 13 *Package Diagram*

i. *Deployment Diagram*

Deployment diagram digunakan untuk menunjukan alokasi artefak pada node dalam desain fisik sebuah sistem. Sebuah *deployment diagram* mewakili sebuah gambaran ke dalam struktur artefak suatu sistem. *Deployment diagram* lebih berfokus pada aspek fisik dari *object-oriented system*. Dengan menggunakan *deployment diagram*, gambaran mengenai distribusi komponen dan relasinya dengan sistem maupun *platform* tempat sistem berjalan bisa digambarkan dengan lebih jelas, sehingga dengan menggunakan *deployment diagram*, gambaran akan interaksi antara *software* dan *hardware* pada sistem akan terlihat lebih jelas.



Gambar 2. 14 *Deployment Diagram*

2.2.2.3 Tujuan Penggunaan UML

Terdapat dua tujuan penggunaan UML, diantaranya yaitu :

1. Memodelkan suatu sistem (bukan hanya perangkat lunak) yang menggunakan konsep berorientasi objek.
2. Menciptakan suatu bahasa pemodelan yang dapat digunakan baik oleh manusia maupun mesin [3].

2.2.3 *Image Processing* (Pengolahan Citra)

Istilah citra atau *image* yang pada umumnya digunakan dalam bidang pengolahan citra diartikan sebagai suatu fungsi kontinu dari intensitas cahaya f(x,y) dalam bidang dua dimensi dengan (x,y) menyatukan suatu koordinat dengan nilai f pada setiap titik menyatukan intensitas atau tingkatan kecerahan atau *gray level*. Suatu citra digital adalah suatu citra kontinyu yang diubah kedalam bentuk diskrit, baik koordinat maupun intensitas cahayanya. Kita dapat menganggap suatu citra digital sebagai suatu matriks, dimana indeks baris dan kolomnya menyatakan koordinat sebuah titik pada citra tersebut dan nilai masing-masing elemennya menyatakan intensitas cahaya pada titik tersebut. Suatu titik

pada sebuah citra digital sering disebut sebagai elemen citra, elemen gambar, dan piksel. Pengolahan citra adalah pemrosesan citra, khususnya dengan menggunakan komputer menjadi citra yang kualitasnya lebih baik.

Data atau informasi tidak hanya disajikan dalam bentuk teks, tapi juga dapat berupa gambar, audio (bunyi, suara, musik) dan video. Keempat macam data atau informasi ini sering disebut multimedia. Citra (image) istilah lain untuk gambar sebagai satu komponen multimedia memegang peranan penting sehingga bentuk informasi visual. Citra mempunyai karakteristik yang tidak dimiliki oleh data teks, yaitu citra kaya dengan informasi. Ada sebuah peribahasa yang berbunyi "*a picture is more than a thousand words*" (sebuah gambar akan lebih bermakna dari seribu kata). Maksudnya tentu sebuah gambar dapat memberikan informasi yang lebih banyak dari pada informasi tersebut disajikan dalam bentuk kata-kata (tekstual). Beberapa jenis operasi pengolahan citra adalah sebagai berikut :

1. Modifikasi Kecemerlangan(*Brightness Modification*)

   Mengubah nilai keabuan/warna dari gelap menuju terang atau sebaliknya mengubah citra yang terlalu cemerlang/pucat menjadi gelap.

2. Peningkatan Kontras (*Contast Enhancement*)

   Dengan peningkatan kontras maka titik yang cenderung gelap menjadi lebih gelap dan yang cenderung terang menjadi lebih cemerlang.

3. Negasi

   Operasi untuk mendapatkan citra negatif (*negative image*).

4. Pengabuan (*Grayscale*)

   Merupakan proses konversi citra dengan warna sebenarnya (*true color*) menjadi citra keabuan (*grayscale*).

5. Pengambangan (*Thresholding*)

   Operasi pengambangan digunakan untuk mengubah citra dengan format skala keabuan, yang mempunyai kemungkinan nilai lebih dari 2 ke citra biner yang memiliki 2 buah nilai (yaitu 0 dan 1).

6. Pencerminan (*Flipping*)

   Pencerminan merupakan proses menggambar citra ke bentuk kebalikannya seperti ketika sedang bercermin.

7. Rotasi (*Rotating*)

   Rotasi yaitu proses memutar koordinat citra sesuai derajat yang ditentukan.

8. Pemotongan (*Cropping*)

   Memotong satu bagian dari citra sesuai kebutuhan.

9. Penskalaan (*Scaling*)

   Mengubah ukuran citra menjadi lebih besar atau lebih kecil.

10. Deteksi Tepi (*Edge Detection*)

    Deteksi tepi (*edge detection*) pada suatu citra adalah suatu proses yang menghasilkan tepi-tepi dari obyek-obyek citra.

2.2.4 *Augmented Reality*

Saat perkembangan teknologi semakin meningkat, hal ini juga erpengaruh terhadap bidang *computer vision*. Definisi *computer vision* secara umun adalah

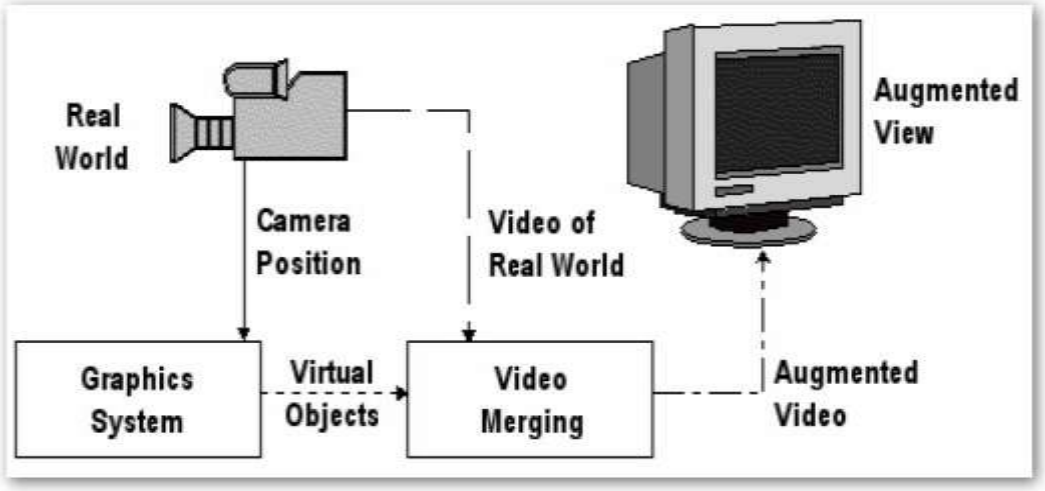
merupakan ilmu dan teknologi bagaimana suatu *machine*/system melihat sesuatu. Masukan untuk suatu system berbasis *computer vision* adalah citra atau *image*. Data citra dapat berbentuk urutan video, citra dari kamera, dan lain-lain.

Beberapa hal yang dikerjakan oleh *computer vision* adalah *recognition, motion, scene reconstruction,* dan *image restoration*. Berikut beberapa contoh penerapan *computer vision*, yaitu *controlling process, detecting events, organizing information, modeling object or environtments,* dan *interaction (human-computer interaction)* [7].

Augmented Reality (AR) adalah sebuah bidang penelitian komputer yang berhubungan dengan kombinasi antara dunia nyata dengan data hasil rekayasa komputer. AR juga salah satu tenologi yang mengunakan teknik *computer vision* dalam menentukan pesesuaian antara citra dan dunia nyata, menghitung pose, *projection matrix*, homografi dari persesuaian-persesuaian ini.

Kunci kesuksesan dari system AR adalah meniru semirip mungin kehidupan dunia nyata. Dengan kata lain, dari sudut pandang pengguna, pengguna tidak perlu belajar terlalu lama dalam menggunakan system AR, sebaliknya dengan cepat mampu mengoperasikan system tersebut berdasarkan pengalaman dalam dunia nyata.

Untuk menjalankan sistem AR, minimal terdiri atas kamera, perangkat monitor, dan dalam kasus-kasus tertentu memerukan perangkat khusus untuk beriteraksi dengan objek virtual.



Gambar 2. 15 Perangkat Pendukung *Augmented Reality*

Perangkat monitor dapat di ganti dengan perangkat *video see-through* untuk meningkatkan kesan impresif dari objek virtual. Perangkat *video see-through*, biasa juga dinamakan *head-mounted display* (HMD), akan memenuhi seluruh sudut pandang pengguna, sehingga kesan nyata dapat tercapai.



Gambar 2. 16 Penggunaan HMD pada sistem *Augmented Reality*



Gambar 2. 17 *Face Tracking*

Macam-macam teknik yang digunakan pada augmented reality diantaranya, yaitu :

1. Teknik Marker *Tracking*

   Teknik ini menggunakan marker untuk mendeteksi posisi tampilnya objek. Marker adalah pola yang dibuat dalam bentuk gambar yang telah dicetak dengan printer yang akan dikenali oleh kamera. Marker yang dapat dideteksi hanya yang berwarna hitam putih seperti gambar di bawah ini :



Gambar 2. 18 Marker *Tracking*

SDK atau *Engine* yang dapat digunakan untuk teknik ini adalah :

a. ARToolkit

b. FLARToolkit

c. SLARToolkit

2. Teknik *Markerless Tracking*

   Teknik masih menggunakan marker sebagai alat untuk mendeteksi. Namun marker yang digunakan dapat berbentuk apa saja, misalnya foto atau bahkan benda. SDK atau Engine yang dapat digunakan untuk teknik ini adalah :

   a. osgART
   b. Layar
   c. D'Fusion Studio
   d. IN2AR

3. Teknik *GPS Tracking*

   Teknik GPS Tracking memanfaatkan lokasi *longitude* dan *latitude* sebagai alat untuk mendeteksi. Biasanya teknik ini digunakan di perangkat lunak untuk *smartphone*. SDK atau Engine yang dapat digunakan untuk teknik ini adalah :

   a. Layar
   b. Junaio
   c. Mixare
   d. KHARMA

4. Teknik *Face Tracking*

   Teknik ini menggunakan wajah manusia untuk dideteksi dan pemanfaatannya bisa untuk hanya untuk *face tracking* maupun *face recognition*. SDK atau Engine yang dapat digunakan untuk teknik ini adalah:

a. Beyond Reality Face
b. Viewdle
c. SHORE

2.2.5 Cara Kerja Algoritma Haar Cascade Classifier

Proses deteksi adanya citra wajah dalam sebuah gambar dapat menggunakan sebuah metode yang dipublikasikan oleh Paul Viola dan Michael Jones tahun 2001. Umumnya disebut algoritma Haar Cascade Classifier. Metode ini merupakan metode yang menggunakan statistik model (*classifier*). Di bawah ini merupakan proses terjadinya deteksi wajah.

2.2.5.1 Penskalaan ukuran gambar

Scaling atau penskalaan merupakan proses mengubah ukuran gambar digital menjadi lebih besar ataupun lebih kecil. Untuk sistem akan memanfaatkan *package display object* yang merupakan *package class* milik Flash. *Scaling* dapat dilakukan dengan memanipulasi ukuran tampilan objek dalam dua cara, menggunakan salah satu properti (*width and height*) atau properti method skala yaitu *scalex* dan *scaley*. Properti skala mewakili ukuran relatif dari tampilan objek dibandingkan dengan ukuran aslinya.

Pada awalnya harus ditentukan dahulu faktor skala untuk membuat layar menampilkan dan memproses gambar dari kamera sesuai dengan ukuran yang diinginkan. Tahap *Scaling* ini membutuhkan skala yang tepat, karena tahap scaling dilakukan pada beberapa proses dan semua nilai skala harus konsisten. Skala yang tepat dan umum digunakan adalah empat karena menyesuaikan

dengan panjang dan lebar layar yang umum digunakan yaitu 500x450, akan tetapi bisa bersifat dinamis dengan mengubah nilai dari variabel faktor skala. Jika faktor skala terlalu besar tidak hanya mempengaruhi terhadap tampilan tetapi juga proses akan menjadi lebih lama. Jika faktor skala terlalu kecil maka penampilan *stage* di flash akan tidak jelas. Fungsi skala dapat dilihat pada kelas HaarCascade di bawah ini.

```
public function set scale(s:Number):void
            {
                  if( s == _scale ){ return; }
                  _scale = s;
                  // update rect's width, height, weight
                  var feature:FeatureBase;
                  inv_window_area  =  1  /  (  base_window_w  *
base_window_h * s * s );

                  var tree:FeatureTree = firstTree;
                  while ( tree != null )
                  {
                        feature = tree.firstFeature;
                        while( feature != null )
                        {
                              feature.setScaleAndWeight(     s,
inv_window_area );
                              feature = feature.next;
                        }
                        tree = tree.next;
                  }
            }
```

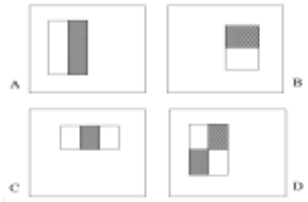
2.2.5.2 Haar *Feature*

Haar *Feature* adalah fitur yang digunakan oleh Viola dan Jones didasarkan pada *Wavelet* Haar. *Wavelet* Haar adalah gelombang tunggal bujur sangkar (satu interval tinggi dan satu interval rendah) . Untuk dua dimensi, satu terang dan satu gelap. Selanjutnya kombinasi-kombinasi kotak yang digunakan untuk pendeteksian objek visual yang lebih baik. Ada tiga tipe kotak (rectangular) fitur pada umunya yaitu:

a. Tipe *two-rectangle feature* (horisontal/vertikal)

b. Tipe *three-rectangle feature*

c. Tipe *four-rectangle feature*



Gambar 2. 19 Haar *Feature*

Adanya fitur Haar ditentukan dengan cara mengurangi rata-rata piksel pada daerah gelap dari rata-rata piksel pada daerah terang. Jika nilai perbedaannya itu diatas nilai ambang atau *threshold*, maka dapat dikatakan bahwa fitur tersebut ada.



Gambar 2. 20 Contoh Penerapan Haar *Feature*

Di dalam *library* Marilena *haar like feature* berada di kelas FeatureBase dimana kelas tersebut merupakan *template* yang dapat diturunkan sesuai dengan jenis fitur yang digunakan. Misalnya dalam sistem ini akan menggunakan fitur dengan jenis b pada gambar 2.21. Maka kelas featureBase dapat diturunkan menjadi kelas Feature2Rects yang memiliki dua variabel HaarRect. Begitu juga jika menggunakan tiga haar maka dapat dibuat kelas Feature3Rect yang memiliki

tiga variabel Haarrect. Berikut ini adalah kelas FeatureBase yang digunakan pada *library* Marilena :

```
public class FeatureBase{
            public  var threshold :Number;
            public  var left_val  :Number;
            public  var right_val :Number;
            public var next:FeatureBase;
            public  function  FeatureBase(  _th:Number,  _lv:Number,
_rv:Number)
            {
                  threshold = _th;
                  left_val  = _lv;
                  right_val = _rv;
            }

            public   function   getSum(   targetImage:TargetImage,
x:int, y:int ):Number{
                  return 0;
            }

            public                                          function
setScaleAndWeight(s:Number,w:Number):void{
            }
      }
```

Nilai dari Haar *Feature* adalah perbedaan antara jumlah nilai-nilai piksel *gray* level dalam daerah kotak hitam dan daerah kotak putih. Kotak Haar *Feature* dapat dihitung secara cepat menggunakan *integral image*.

2.2.5.3 Integral Image

Integral Image digunakan untuk menentukan ada atau tidak fitur Haar pada sebuah gambar dan pada skala yang berbeda secara efisien. Pada umumnya, pengintegrasian tersebut berarti menambahkan unit-unit kecil secara bersamaan. Dalam hal ini unit-unit kecil tersebut adalah nilai-nilai piksel. Nilai integral untuk masing-masing piksel adalah jumlah dari semua piksel-piksel dari atas sampai bawah. Dimulai dari kiri atas sampai kanan bawah, keseluruhan gambar itu dapat

dijumlahkan dengan beberapa operasi bilangan bulat per piksel. Berikut contoh gambar yang dapat dijadikan perhitungan *integral image*.



Gambar 2. 21 Contoh Bagian Dari Haar *Feature*

Sedangkan di bawah ini merupakan contoh perhitungan piksel wajah dari kiri atas sampai kanan bawah. Nilai *integral image* dari kolom kesatu baris kesatu tetap, sedangkan *integral image* kolom kedua baris kesatu merupakan jumlah nilai piksel wajah di sebelah kirinya. Kolom kesatu baris kedua memperoleh nilai integral image dari jumlah nilai piksel kolom kesatu baris kesatu. Di bawah ini perhitungan lengkapnya :

1	2	3	1	1+2	1+2+3
2	3	4	1+2	1+2+2+3	1+2+3+2+3+4
4	5	1	1+2+4	1+2+2+3+4+5	1+2+3+2+3+4+4+5+1

1	3	6
3	8	15
7	**17**	**25**

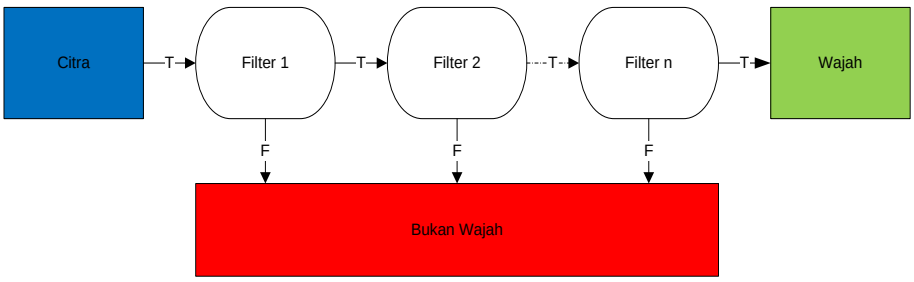
Dari perhitungan *integral image* diatas, maka diperolah nilai integral fitur putih = {3,8,15} dan nilai integral fitur hitam = {7,17,25}. Maka nilai *haar feature* tersebut adalah :

f(x) = (7 + 17 + 25) - (3 + 8 + 15) = 23

Proses perhitungan *integral image* di dalam sistem ini terdapat pada kelas TargetImage dalam *library* Marilena.

2.2.5.4 Cascade Classifier

Cascade classifier adalah sebuah rantai *stage classifier*, dimana setiap *stage classifier* digunakan untuk mendeteksi apakah didalam image *sub window* terdapat obyek yang diinginkan (*object of interest*). Pada proses ini menggunakan klasifikasi bertingkat. Filter pada masing-masing level mengklasifikasikan gambar yang sebelumnya telah difilter. Ketika filter berhasil melewatkan *image region*, *image region* kemudian masuk pada filter yang selanjutnya. *Image region* yang telah melalui semua filter akan dianggap sebagai "Wajah" dan jika filter tersebut gagal, maka daerah tersebut dianggap sebagai "Bukan Wajah".



Gambar 2. 22 Cascade Classifier

Untuk dapat menentukan wajah atau bukan wajah, Haar menggunakan sebuah pelatihan. Proses pelatihan ini dikenal dengan algoritma *haartraining* yang pada akhir pelatihan akan menghasilkan parameter model statistik.

Training ini memiliki isi yang berbeda baik dari segi jumlah *stage*, jumlah *tree*, model segi empat dari fitur (*rectangle*) maupun nilai *treshold* nya. *Stage* melambangkan banyaknya tingkatan dalam *cascade of classifier*, dalam *training*

ini digunakan 22 tingkatan (stage 0 sampai stage 21). Tingkatan ini digunakan untuk mengurangi jumlah *sub window* citra yang perlu diperiksa. Di tingkatan pertama dilakukan pengklasifikasian terhadap seluruh *sub window* citra, lalu di tingkatan kedua dilakukan pengklasifikasian terhadap *sub window* yang berasal dari hasil pengklarifikasian tingkatan pertama. Maka semakin tinggi tingkatanya semakin sedikiit jumlah *sub window* yang harus diperiksa. Di tiap tingkatan terdiri dari beberapa *tree*, biasanya semakin tinggi tingkatan maka *tree* yang terdapat di dalamnya pun semakin banyak. Pada *stage* 0 terdapat 2 *tree* dan pada *stage* 21 terdapat 212 *tree*. Setelah proses pendeteksian wajah berhasil dilakukan, maka sistem akan menetukan wajah dan tidak. Di bawah ini merupakan *capture* dari *stage* 0 tingkatan *tree* pada kelas HaarCascade untuk proses deteksi wajah.

```
tree = firstTree = new FeatureTree(0.822689414024353);
                  feature     =     tree.firstFeature     =     new
Feature2Rects(0.004014195874333382,0.0337941907346249,0.8378106951
713562,[3,7,14,4,-1],[3,9,14,2,2]);
                  feature      =      feature.next      =      new
Feature2Rects(0.0151513395830989,0.1514132022857666,0.748881220817
5659,[1,2,18,4,-1],[7,2,6,4,3]);
                  feature.next                =                new
Feature2Rects(0.004210993181914091,0.0900492817163467,0.6374819874
763489,[1,7,15,9,-1],[1,10,15,3,3]);
```

Dari 22 tingkatan *tree* di atas, proses deteksi wajah dapat diketahui melalui pseudo code dari algoritma Haar Cascade Classifier di bawah ini.

```
1.  function run (r: Rectangle, tree: FeatureTree) ← integer
2.  (I.S : nilai rectangle dari strong classifier dan tree telah
    terdefinisi)
3.  (F.S : menghasilkan classifier terbaik)
4.
5.  Kamus :
6.  x,y,w,h : integer
7.  mean : real
8.  variance_norm_factor, inv_window_area : real
9.  features : array of real
10. val, sum, st_th : real
```

```
11. i,j : integer
12. feature : FeatureBase
13. tree : FeatureTree
14.
15. Algoritma :
16.
17. x ← r↑.x
18. y ← r↑.y
19. w ← r↑.width
20. h ← r↑.height
21. val ← 0
22. sum ← 0
23. i ← 0
24. j ← 0
25. st_th ← 0
26.
27. mean ← targetImage.getSum(x,y,w,h) *  inv_window_area
28. variance_norm_factor ← targetImage.getSum2(x,y,w,h)*
    inv_window_area - mean*mean
29.
30. if(variance_norm_factor >= 0) then
31.    variance_norm_factor ← Math.sqrt(variance_norm_factor)
32. else
33.    variance_norm_factor ← 1
34. endif
35.
36. while (tree ≠ nil) do
37.    feature ← tree.firstFeature
38.    val ← 0
39.    st_th ← tree.stage_threshold
40.
41.    while (feature  ≠ nil) do
42.          sum ← feature.getSum(targetImage, x, y)
43.
44.          if  (sum  <  feature.threshold  *  variance_norm_factor)
    then
45.                val = val + feature.left_val
46.          else
47.                val = val + feature.right_val
48.          endif
49.
50.          if (val > st_th) then
51.                break
52.          endif
53.          feature ← feature.next
54.    endwhile
55.
56.    if (val < st_th) then
57.          return 0
58.    endif
59.    tree ← tree.next
60. endwhile
61. return 1
62. endfunction
```

Pada citra yang mengandung wajah dilakukan proses pemberian kotak di sekitar wajah. Hal ini untuk mengecek apakah proses pendeteksian wajah ini telah berhasil mengenali wajah dengan tepat. Di bawan ini gambar wajah terdeteksi dari hasil Haar *training*.



Gambar 2. 23 Wajah Terdeteksi

2.2.6 *Library Marilena*

Library Marilena merupakan *library* yang mendukung deteksi wajah pada flash dengan menggunakan ActionScript. *Marilena* merupakan *library* yang telah berjalan baik untuk mendeteksi wajah atau benda lain yang berdasarkan algoritma Haar . Hanya saja kekurangan dari Haar yang digunakan adalah tidak dapat mendeteksi wajah pada posisi miring atau selain posisi tegak lurus. Sebenarnya *Marilena* pada dasarnya berasal dari library OpenCV yang digunakan pada bahasa C, yang di konversi ke ActionScript. Seorang pria berkebangsaan Jepang bernama Ohtsuka Masakazu yang telah menciptakan *Marilena*.

Baru-baru ini seorang pria bernama Mario Klingemann dari Quasimondo.com telah membuat beberapa perubahan untuk mengoptimasikan

Marilena versi aslinya. Untuk mengunduh Marilena dapat menggunakan svn dengan memasukkan alamat http://www.libspark.org/svn/as3/Marilena/trunk/.

Pada aplikasi virtual aksesories ini menggunakan empat *library object detection* dalam *marilena* yaitu ObjectDetector, ObjectDetectorEvent, ObjectDetectorOptions, dan CameraBitmap. Keempat *library* tersebut dipanggil dalam kelas FaceDetector_Camera yang berfungsi untuk menampilkan wajah dan menampilkan objek virtual pada layar kamera webcam. Di bawah ini ActionScript dari empat *library* yang digunakan diantaranya, yaitu :

1. ObjectDetector

   Berfungi sebagai kelas pertama untuk mendeteksi sebuah objek.

2. ObjectDetectorEvent

   Berfungsi sebagai event atau kejadian jika terdapat objek yang terdeteksi.

3. ObjectDetectorOptions

   Berfungsi sebagai pilihan objek yang akan dipilih. Pada marilena objek yang dipilh yaitu pada area wajah, jadi pada kelas ini objek wajah yang akan terdeteksi.

4. CameraBitmap

   Berfungsi untuk mengatur ukuran dari kamera yang akan ditampilkan di layar dan juga berfungsi untuk mengaktifkan kamera.

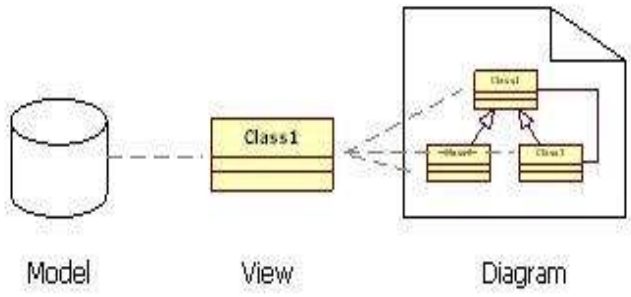
2.2.7 StarUML

StarUML adalah software permodelan yang mendukung UML (UnifiedModeling Language). Berdasarkan pada UML version 1.4 dan dilengkapi 11 macam diagram yang berbeda, mendukung notasi UML 2.0 dan juga

mendukung pendekatan MDA (Model Driven Architecture) dengan dukungan konsep UML.StarUML dapat memaksimalkan pruduktivitas dan kualitas dari suatu software project. Berikut ini merupakan konsep dasar dari StarUml :

2.2.7.1 Model, View, dan Diagram

StarUML membuat perbedaan konseptual yang lebih jelas antara models, views and diagrams. Model adalah elemen yang memuat informasi untuk model software. View adalah suatu ekpresi visual dari informasi di dalam model dan Diagram adalah suatu koleksi dari elemen yang memberikan pemikiran user didalam mendesaian secara spesifik.



Gambar 2. 24 *Model, View,* dan *Diagram*

2.2.7.2 *Project*

1. ***Project***

   Project adalah unit manajemen dasar didalam StarUML. Suatu *project* dapat mengatur satu atau lebih model *software*. *Project* merupakan top-

level package yang selalu ada di dalam beberapa model *software*. Secara umum, satu *project* disimpan dalam satu *file.*

2. ***File Project***

   File project disimpan kedalam format XML dengan extension ".UML" . Semua model,view dan diagram yang dibuat dengan StarUML disimpan dalam satu *file project*. *File project* berisikan informasi sebagai berikut :

   a. UML profiles used in the project

   b. Unit files referenced by the project

   c. Information for all models contained in the project

   d. Information for all diagrams and views contained in the project

3. ***Module***

   Modul adalah suatu *package* yang menyediakan fungsi – fungsi baru dan feature sebagai perluasan dari StarUML. Modul dapat dibuat sebagai kombinasi dari beberapa elemen – elemen *extension* dan juga membuat beberapa jenis elemen – elemen didalam suatu modul.

2.2.8 *Adobe Flash*

Adobe Flash (dahulu bernama Macromedia Flash) adalah salah satu perangkat lunak komputer yang merupakan produk unggulan Adobe *Systems*. Adobe Flash digunakan untuk membuat gambar vektor 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 Macromedia. Flash 1.0 diluncurkan pada tahun 1996 setelah Macromedia membeli program animasi vektor bernama FutureSplash. Versi terakhir yang diluncurkan di pasaran dengan menggunakan nama 'Macromedia' adalah adalah Macromedia Flash 8. Pada tanggal 3 Desember 2005 Adobe Systems mengakuisisi Macromedia dan seluruh produknya, sehingga nama Macromedia Flash berubah menjadi 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 efek animasi 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 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*, karena mempunyai keunggulan antara lain kecil dalam ukuran *file* outputnya.

Movie-movie Flash memiliki ukuran *file* yang kecil dan dapat ditampilkan dengan ukuran layar yang dapat disesuaikan dengan keingginan. Aplikasi Flash merupakan sebuah standar aplikasi industri perancangan animasi *web* dengan peningkatan pengaturan dan perluasan kemampuan integrasi yang lebih baik. Banyak fitur-fitur baru dalam Flash yang dapat meningkatkan kreativitas dalam pembuatan isi media yang kaya dengan memanfaatkan kemampuan aplikasi tersebut secara maksimal. Fiture-fiture baru ini membantu kita lebih memusatkan perhatian pada desain yang dibuat secara cepat, bukannya memusatkan pada cara kerja dan penggunaan aplikasi tersebut. Flash juga dapat digunakan untuk mengembangkan secara cepat aplikasi-aplikasi *web* yang kaya dengan pembuatan *script* tingkat lanjut. Di dalam aplikasinya juga tersedia sebuah alat untuk men-*debug script*. Dengan menggunakan *code hint* untuk mempermudah dan mempercepat pembuatan dan pengembangan isi ActionScript secara otomatis.



Gambar 2. 25 Logo Adobe Flash

2.2.9 *Action Script*

ActionScript merupakan bahasa pemrograman yang bekerja di dalam *platform* Adobe Flash. ActionScript adalah menunjukkan koleksi set dari *action, function, event,* dan *event handler* yang memungkinkan dikembangkan oleh para *developer* untuk membuat *flash movie* yang lebih komplek dan lebih interaktif.

ActionScript merupakan bahasa pemrograman berorientasi objek didasarkan pada ECMAScript-standar yang sama yang menjadi dasar JavaScript dan ditujukan untuk berjalan di atas sebuah runtime environment (seperti Flash Player, Adobe AIR, atau Flash Lite). Mirip seperti prinsipnya Java, Flash pada dasarnya berjalan diatas sebuah AVM (ActionScript Virtual Machine).

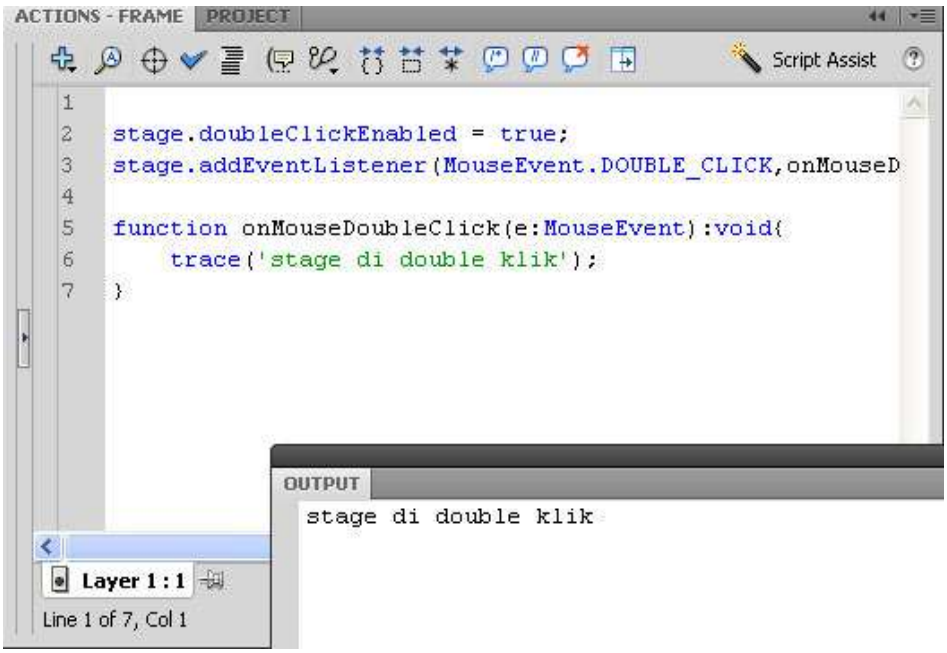
ActionScript sendiri telah memiliki beberapa versi dari satu sampai kini ada versi tiga. ActionScript 2 yaitu versi ActionScript yang telah digunakan dalam Flash Player 8 dan sebelumnya, tetap didukung dalam Flash Player 9 dan Flash Player 10. Pada akhirnya tahun 2006 Adobe merilis Action Script versi 3.0, yang merupakan bahasa pemprograman yang bisa didapatkan di Adobe Flash CS3 ke atas beda halnya dengan Macromedia Flash 8 dan sebelumnya, yang hanya mendukung action script 2.

Di dalam Flex *framework,* antarmuka pengguna dibuat menggunakan *file* MXML, sedangkan ActionScript 3 digunakan hanya untuk menciptakan file akhir SWF. Oleh karena itu ActionScript 3.0 memiliki beberapa keuntungan dibandingkan versi pendahulunya diantaranya yaitu :

a. Memiliki lebih banyak laporan detail error yang ada. Hal tersebut dikarenakan ActionScript 3.0 mewajibkan penulisan tipe data dari variabel, argument, fungsi, dan yang lainnya. Berbeda dengan ActionScript 2 yang menjadikan hal tersebut sebagai *optional*. Perubahan tersebut tentu dapat mempermudah *error checking.*

b. Perubahan sintak yang terjadi ActionScript 3.0 seperti pada nama properti yang di awali *underscores* pada ActionScript 2.0 dan tidak perlu lagi digunakan pada ActionScript 3.0.

c. Tampilan Arsitektur yang baru. Banyak metode sebelumnya untuk menambahkan sesuatu secara dinamis, sekarang memiliki proses yang lebih sederhana dan juga lebih mudah.



Gambar 2. 26 Contoh ActionScript 3 Mouse Event

2.2.10 Adobe Photoshop CS

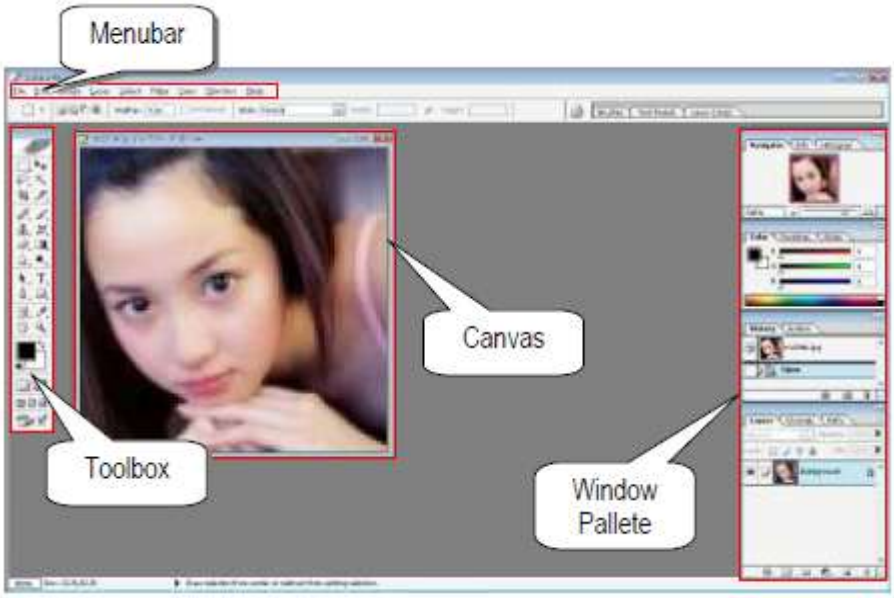
Photoshop adalah software yang digunakan untuk memodifikasi gambar atau foto secara profesional baik meliputi modifikasi obyek yang sederhana maupun yang sulit sekalipun. Photoshop merupakan salah satu *software* yang berguna untuk mengolah gambar berbasis bitmap, yang mempunyai tool dan efek yang lengkap, sehingga dapat menghasilkan gambar atau foto yang berkwalitas

tinggi (jika ingin lebih jauh mengetahui tentang gambar berbasis *bitmap* silakan download dokumennya di sini) [6].

Kelengkapan fitur yang ada di dalam Photoshop inilah yang akhirnya membuat software ini banyak digunakan oleh desainer grafis profesional. Dan mungkin juga sampai saat ini masih belum ada *software* desain grafis lain yang bisa menyamai kelengkapan fitur dalam Photoshop.

2.2.10.1 Bagian-Bagian Adobe Photoshop CS

Gambar 2.35 dibawah ini merupakan gambar dari bagian-bagian yang ada pada Adobe Photoshop CS [6].



Gambar 2. 27 Bagian-bagian Adobe Photoshop CS

Keterangan :

a. *Menu Bar*

Menu bar adalah menu pulldown yang berisi perintah-perintah dalam Photoshop seperti menu *File*, *Edit*, *Image*, *Layer*, *Select*, *Filter*, *View*, Window, dan Help. Baris menu ini terkelompok berdasarkan topiknya.

b. *Toolbox*

Toolbox adalah alat-alat yang digunakan untuk memodifikasi *image* (gambar atau foto). Alat-alat ini juga dikelompokkan menurut jenisnya.

c. *Canvas*

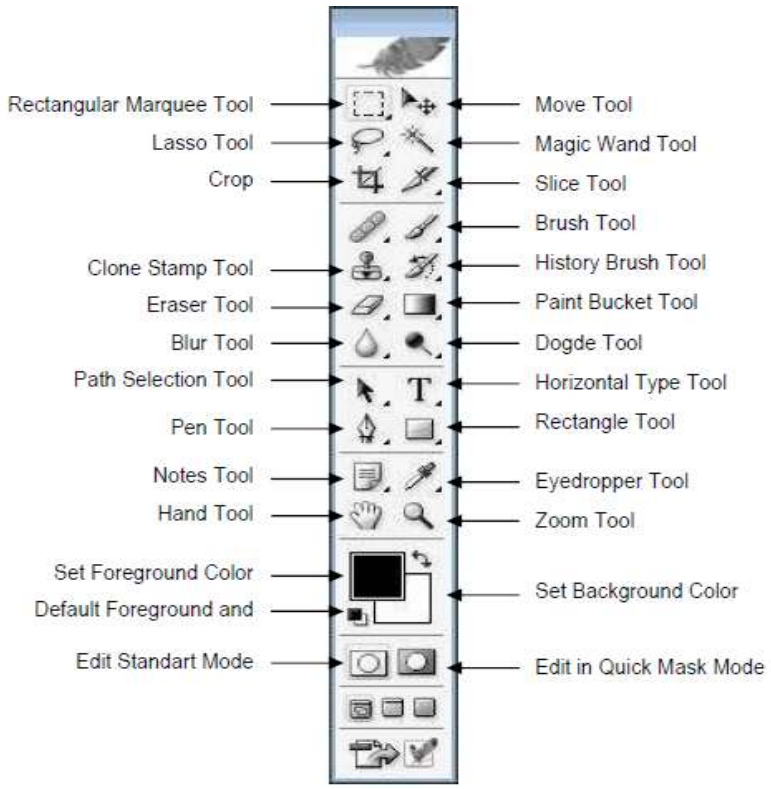
Canvas adalah bidang yang digunakan sebagai tempat untuk meletakkan *image*. Biasanya ukuran *canvas* akan sama dengan ukuran image, tetapi dalam Photoshop kita dapat merubah ukuran *canvas* dan *image* sesuai dengan kebutuhan. Kalau kita memunculkan canvas baru biasanya ada tiga pilihan yaitu canvas yang putih, berwarna dan transparan.

d. *Window Pallete*

Window pallete adalah window yang berguna untuk memilih atau mengatur berbagai parameter pada saat menyunting *image* dalam Photoshop. Untuk menampilkan *Window Pallete* dapat kita lakukan dengan cara memilih menu Window kemudian pilih *pallete* yang dimunculkan [6].

2.2.10.2 Toolbox Pada Adobe Photoshop CS

Gambar 2.36 pada halaman berikutnya ini merupakan bagian-bagian dari *toolbox* yang ada pada Adobe Photoshop CS [6].



Gambar 2. 28 Toolbox

Keterangan :

a. *Rectangular Marquee Tool*

Digunakan untuk menyeleksi atau mem-blok bagian dari suatu *image* dengan bentuk seleksi persegi. Di dalamnya ada *Elliptical Marquee Tool*, *Single Row*, *Marquee Tool*, dan *Single Column Marquee Tool*.

b. *Lasso Tool*

Digunakan untuk menyeleksi atau mem-blok bagian dari suatu image dengan bentuk bebas dengan cara mendrag bagian image yang akan diseleksi. Di dalamnya ada *Poligonal Lasso Tool*, dan *Magnetic Lasso Tool*.

c. *Crop*

Digunakan untuk menyeleksi bagian dari suatu image dengan bentuk seleksi persegi dan membuang bagian yang tidak diinginkan.

d. *Clone Stamp Tool*

Digunakan untuk mengambil contoh dari suatu image dan membuat kopinya dimanapun.

e. *Eraser Tool*

Digunakan untuk menghapus *image*. Ini juga dapat digunakan untuk mengembalikan sebuah *image* menjadi sebuah daerah awal.

f. *Blur Tool*

Digunakan untuk mengaburkan bagian dari suatu *image*.

g. *Path Selection Tool*

Digunakan untuk mempercerah bagian dari suatu *image*.

h. *Pen Tool*

Digunakan untuk merancang atau mengedit *path*.

i. *Notes Tool*

Digunakan untuk membuat catatan pada suatu bagian *image*.

j. *Hand Tool*

Digunakan untuk menggeser *image* jika lebar *image* melebihi ukuran *window*.

k. *Set Foreground Color*

Digunakan sebagai indikator warna foreground yang aktif.

l. *Default Foreground and Background Color*

Digunakan untuk mengembalikan warna standar yaitu *foreground* hitam dan *background* putih.

m. *Edit Standart Mode*

Digunakan untuk membuat seleksi dan menampilkan standar *painting* dan *editing*.

n. *Move Tool*

Digunakan untuk memindah bagian dari *image* yang di blok atau layer diantara *image* atau *image* lainnya.

o. *Magic Wand Tool*

Digunakan untuk membuat seleksi atau blok secara otomatis yang hampir sama dengan warna areanya.

p. *Slice Tool*

Digunakan untuk membuat atau memblok setiap area *rectangular* dari sebuah *image*.

q. *Brush Tool*

Digunakan untuk menggambar dengan warna *foreground* dengan menggunakan standar *brushes* atau custom *brushes*.

r. *History Brush Tool*

Digunakan untuk menggambar di atas sebuah *image* dengan daerah sebelumnya yang telah ditentukan dalam rangka membenarkan kesalahan atau untuk menggambar di atas sebuah *image*.

s. *Paint Bucket Tool*

Digunakan untuk memberi warna suatu bagian dari image dengan warna *foreground*.

t. *Dogde Tool*

Digunakan untuk mencerahkan bagian dari suatu *image*.

u. *Horizontal Type Tool*

Digunakan untuk membuat teks pada suatu *image*.

v. *Rectangle Tool*

Digunakan untuk menggambar kotak pada suatu *image*.

w. *Eyedropper Tool*

Digunaan untuk memilih warna *foreground* maupun *background* dari warna yang ada pada sebuah *image*.

x. *Zoom Tool*

Digunakan untuk memperbesar atau memperkecil tampilan sebuah *image*. Perubahan tampilan menggunakan *zoom tool* ini tidak mempengaruhi ukuran *pixel* dalam sebuah *image*.

y. *Set Background Color*

Digunakan sebagai indikator warna *background* yang aktif dan sekaligus dapat digunakan untuk mengganti warna *background* dengan warna yang lain.

z. *Edit in Quick Mask Mode*

Digunakan untuk merancang dan mengedit *selection mask* sementara [6].

2.2.11 Kuesioner dan Skala [12]

Kuesioner adalah suatu teknik pengumpulan informasi yang memungkinkan analis mempelajari sikap-sikap, keyakinan, perilaku, dan karakteristik beberapa orang utama di dalam organisasi yang bisa terpengaruh oleh sistem yang diajukan atau oleh sistem yang sudah ada. Dengan menggunakan kuesioner, analis berupaya mengukur apa yang ditemukan dalam wawancara, selain itu juga untuk menentukan seberapa luas atau terbatasnya sentimen yang diekspresikan dalam suatu wawancara.

Penggunaan kuesioner tepat apabila :

a. Responden (orang yang merenpons atau menjawab pertanyaan) saling berjauhan.
b. Melibatkan sejumlah orang di dalam proyek sistem, dan berguna bila mengetahui berapa proporsi suatu kelompok tertentu yang menyetujui atau tidak menyetujui suatu fitur khusu dari sistem yang diajukan.
c. Melakukan studi untuk mengetahui sesuatu dan ingin mencari seluruh pendapat sebelum proyek sistem diberi petunjuk-petunjuk tertentu.
d. Ingin yakin bahwa masalah-masalah dalam sistem yang ada bisa diidentifikasi dan dibicarakan dalam wawancara tindak lanjut.

Jenis-jenis pertanyaan di dalam kuesioner yaitu, meliputi :

a. Pertanyaan Terbuka

   Pertanyaan-pertanyaan yang memberi pilihan-pilihan respons terbuka kepada responden. Pada pertanyaan terbuka antisipasilah jenis respons yang muncul. Respons yang diterima harus tetap bisa diterjemahkan dengan benar.

b. Pertanyaan Tertutup

Pertanyaan-pertanyaan yang membatasi atau menutup pilihan-pilihan respons yang tersedia bagi responden.

Untuk memperoleh data dari responden, maka diperlukan skala yang dapat digunakan untuk mengukur sikap yang menjadi dasar kepribadian suatu populasi. Para ahli sosiologi membedakan dua tipe skala menurut fenomena soasia yang diukur, yaitu :

1. Skala pengukuran untuk mengukur perilaku susila dan kepribadian.
2. Skala pengukuran untuk mengukur berbagai aspek budaya lain dan lingkungan sosial.

Berbagai skala yang dapat digunakan untuk penelitian Administrasi antara lain:

1. Skala *Likert*

   Skala *Likert* digunakan untuk mengukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang tentang fenomena sosial. Dengan Skala *Likert*, variabel yang akan diukur dijabarkan menjadi indikator variabel. Kemudian indikator tersebut dijadikan sebagai titik tolak untuk menyusun item-item instrumen yang dapat berupa pertanyaan atau pernyataan. Berdasarkan data hasil kuesioner, dapat dicari persentase masing-masing jawaban dengan menggunakan rumus : Y = P/Q * 100%

   Keterangan :

   P = Jumlah skor

   Q = Jumlah responden x jumlah skor

Y = Nilai persentase

Di bawah ini adalah hasil persentasi masing-masing jawaban yang sudah dihitung nilainya dengan menggunakan rumus diatas. Kuesioner ini diujikan kepada 10 orang dengan skala lima menggunakan jenis skala *likert*.

Berikut ini adalah skor yang diberikan untuk jawaban kuesioner :

1. Jawaban SS diberi nilai 5
2. Jawaban S diberi nilai 4
3. Jawaban CS diberi nilai 3
4. Jawaban KS diberi nilai 2
5. Jawaban TS diberi nilai 1

Keterangan :

1. SS : Sangat Setuju
2. S : Setuju
3. CS : Cukup Setuju
4. KS : Kurang Setuju
5. TS : Tidak Setuju

2. *Skala Guttman*

Skala pengukuran dengan tipe ini akan didapatkan jawaban yang tegas. diantaranya : ‘ya’ dan ‘tidak’; ‘benar-salah’, dan lain-lain. Data yang diperoleh dapat berupa data interval atau rasio dikhotomi (dua alternatif). Jadi, kalau pada Skala *Likert* terdapat 1,2,3,4,5 interval, dari kata ‘sangat setuju’ sampai ‘sangat tidak setuju’, maka pada Skala Guttman hanya ada dua interval yaitu ‘setuju’ atau

‘tidak setuju’. Penelitian menggunakan Skala Guttman dilakukan bila ingin mendapatkan jawaban yang tegas terhadap suatu permasalahan yang ditanyakan.

3. Semantic Differential

Skala pengukuran yang berbentuk Semantic Differensial dikembangkan oleh Osgood. Skala ini juga digunakan untuk mengukur sikap, hanya bentuknya tidak pilihan ganda maupun checklist, tetapi tersusun dalam satu garis kontinu yang jawaban “sangat positifnya” terletak di bagian kanan garis, dan jawaban “sangat negatif” terletak di bagian kiri garis, atau sebaliknya. Data yang diperoleh adalah data interval, dan biasanya skala ini digunakan untuk mengukur sikap/karakteristik tertentu yang dipunyai oleh seseorang.

4. Rating Scale

Dari ke tiga skala pengukuran seperti yang telah dikemukakan, data yang diperoleh semuanya adalah data kualitatif yang kemudian dikuantitatifkan. Tetapi dengan *Rating Scale*, data mentah yang diperoleh berupa angka kemudian ditafsirkan dalam pengertian kualitatif.

Responden menjawab, senang atau tidak senang, setuju atau tidak setuju, pernah atau tidak pernah adalah merupakan data kualitatif. Dalam skala model *Rating Scale*, responden tidak akan menjawab salah satu dari jawaban kualitatif yang telah disediakan, tetapi menjawab salah satu jawaban kuantitatif yang telah disediakan. Oleh karena itu Rating Scale ini lebih fleksibel, tidak terbatas untuk pengukuran sikap saja tetapi untuk mengukur persepsi responden terhadap fenomena lainnya, seperti skala untuk mengukur status sosial ekonomi, pengetahuan, kemampuan, dan lain-lain.

Yang penting dalam Rating Scale adalah harus dapat mengartikan setiap angka yang diberikan pada alternatif jawaban pada setiap item instrumen. Orang tertentu memilih jawaban angka 2, tetapi angka 2 oleh orang tertentu belum tentu sama maknanya dengan orang lain yang juga memilih jawaban dengan angka 2.