# BAB 2
# TINJAUAN PUSTAKA

## 2.1 Profil Objek Penelitian

Objek penelitian untuk tugas akhir ini adalah Sekolah Dasar Negeri Cicadas 2 Bandung, yang bertempat di Jl. Cikutra no.58 Kelurahan Padasuka, Kecamatan Cibeunying Kidul, Kota Bandung.

### 2.1.1 Visi

Menjadikan lembaga sebagai mitra terpercaya masyarakat dalam memberikan pelayanan pendidikan untuk membentuk peserta didik yang berkarakter berlandaskan iman dan takwa.

### 2.1.2 Misi

Untuk mencapai visi sekolah tersebut perlu merumuskan misi sekolah yang dikelompokkan dalam kegiatan jangka pendek dan jangka panjang dengan arah yang jelas dan sistematis. Berikut rumusan misi Sekolah Dasar Negeri Cicadas 2 :

a. Menyiapkan peserta didik yang unggul dibidang ilmu pengetahuan sebagai bekal untuk melanjutkan pendidikan ke jenjang yang lebih tinggi**.**
b. Menumbuhkembangkan potensi peserta didik yang sesuai dengan bakat dan minat secara optimal.
c. Membentuk peserta didik yang berkarakter, sehingga terbangun manusia yang memiliki kepribadian jujur, bertanggung jawab, disiplin, berbudi pekerti luhur dan berakhlak mulia.

### 2.1.3 Tujuan

Tujuan dari Sekolah Dasar Negeri Cicadas 2 adalah :

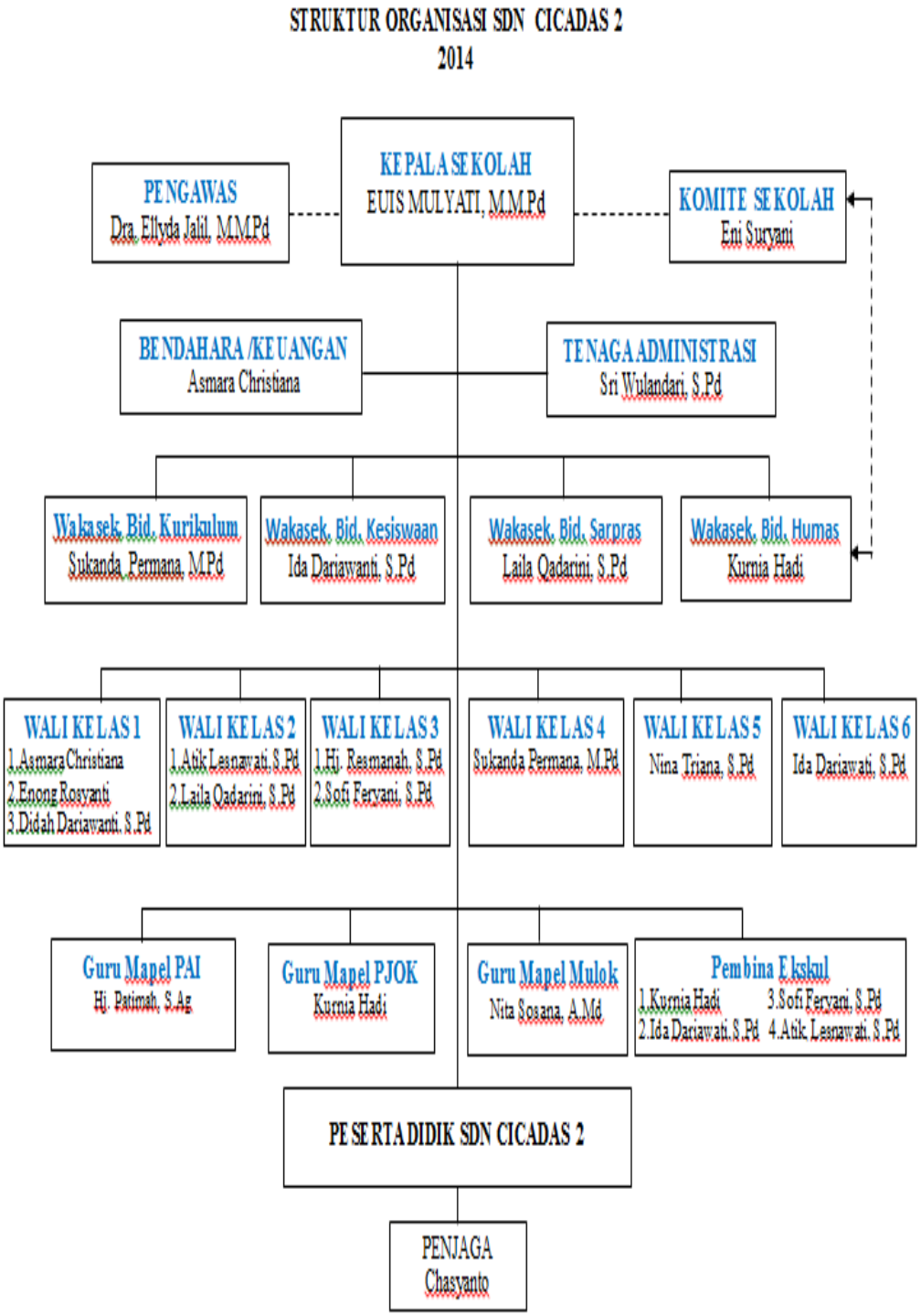
a. Agar peserta didik dari seluruh tingkatan mampu bersaing dalam setiap event seperti penerimaan siswa baru SMP Negeri, lomba calistung, olimpiade MIPA dan event O2SN.
b. Agar peserta didik dapat memiliki sikap peduli dan sikap toleransi di lingkungan sekolah, keluarga dan masyarakat.
c. Agar peserta didik mampu mengaktualisasikan nilai-nilai kecerdasan intelegensi, kecerdasan emosi, dan kecerdasan emosional di lingkungan sekolah, keluarga dan masyarakat dalam kehidupan sehari-hari.

### 2.1.4 Identitas SDN Cicadas 2

**Tabel 2.1 Identitas SD Negeri Cicadas 2**

| | |
|---|---|
| 1. Nama Sekolah | : SD Cicadas 2 |
| 2. Status Sekolah | : Negeri |
| 3. Alamat | : Jl. Cikutra no.58 Kelurahan Padasuka Kecamatan Cibeunying Kidul Kota Bandung |
| 4. Tahun Pendirian | : 1942 |
| 5. Status Tanah | : Tanah Pemerintah |
| 6. Status Bangunan | : Hak Milik |
| 7. Status Akreditasi | : Terakreditasi A |
| 8. Tahun Akreditasi Terakhir | : 2011 |
| 9. NPSN | : 20245153 |
| 10. NSS | : 101026012002 |
| 11. No. NPWP | : 00.5477104423000 |
| 12. Luas Lahan | : 2.045 m² |
| 13. Waktu Penyelenggaraan Pendidikan | : Pagi |

### 2.1.5 Struktur Organisasi SDN Cicadas 2




**Gambar 2.1 Struktur Organisasi SD Negeri Cicadas 2**

## 2.2 Multimedia

Multimedia adalah pemanfaatan komputer untuk membuat dan menggabungkan teks, grafik, audio dan video dengan menggunakan *tool* yang memungkinkan pemakai berinteraksi, berkreasi dan berkomunikasi (Daryanto, 2003).

Secara umum multimedia terbagi menjadi dua kategori, yaitu :

a. Multimedia Linier

Multimedia linier merupakan suatu multimedia yang tidak dilengkapi dengan alat pengontrol apapun yang dapat dioperasikan oleh pengguna. Multimedia ini berjalan sekuensial atau berurutan. Contohnya adalah film yang ada di televisi.

b. Multimedia Interaktif

Multimedia interaktif merupakan suatu multimedia yang dilengkapi dengan alat pengontrol yang dapat dioperasikan oleh pengguna, sehingga pengguna dapat memilih apa yang akan dikehendaki untuk proses selanjutnya. Contohnya adalah multimedia pembelajaran interaktif (Chandra, 2006).

### 2.2.1 Teks

Teks merupakan dasar dari pengolahan kata dan informasi berbasis multimedia. Pada umumnya setiap orang yang biasa menggunakan komputer sudah terbiasa dengan teks. Dalam kenyataannya multimedia menyajikan informasi kepada audiens dengan cepat, karena tidak diperlukan membaca secara rinci dan teliti (Arsyad, 2011).

### 2.2.2 Image

Secara umum *image* atau grafik berarti *still image* seperti foto dan gambar. Manusia sangat berorientasi pada visual dan gambar merupakan sarana yang sangat baik untuk menyajikan informasi (Madcoms, 2011).

### 2.2.3 Audio

Media audio adalah media yang mengandung pesan dalam bentuk auditif yang dapat merangsang pikiran, perasaan, perhatian dan kemauan para siswa untuk mempelajari bahan ajar dan jenisnya. Penyajian audio atau suara merupakan cara lain untuk lebih memperjelas pengertian suatu informasi. Contohnya, narasi merupakan kelengkapan dari penjelasan yang dilihat melalui video. Suara dapat lebih menjelaskan karakteristik suatu gambar, misalnya music dan suara efek. Salah satu bentuk bunyi yang bisa digunakan dalam produksi multimedia adalah *wafeform* audio yang merupakan format *file* audio yabng berbentuk digital. Kualitas produknya bergantung pada sampling rate (banyaknya sampel per detik). *Waveform* (wav) merupakan standar untuk *Windows PC* (Madcoms, 2011).

### 2.2.4 Video

Menurut Chris Crawford, video merupakan elemen multimedia paling kompleks karena penyampaian informasi yang lebih komunikatif dibandingkan gambar biasa. Walaupun terdiri dari elemen-elemen yang sama seperti grafik, suara dan teks, namun bentuk video berbeda dengan animasi. Perbedaan terletak pada penyajiannya. Dalam video, informasi disajikan dalam kesatuan utuh dari objek yang dimodifikasi sehingga terlihat saling mendukung penggambaran yang seakan terlihat hidup.

## 2.3 Animasi

Animasi merupakan sebuah proses merekam dan memainkan kembali serangkaian gambar statis untuk mendapatkan sebuah ilusi pergerakan (Fernandes, 2002). Sejak dahulu manusia mencoba menangkap nuansa dari sebuah pergerakan ke dalam media. Sejarah awal animasi akan membawa kita jauh ke masa mesir kuno. Terbukti dengan ditemukannya rangkaian gerak dalam bentuk gambar pada artefak bangsa mesir kuno 2000 tahun SM (Fernandes, 2002).

Sejalan dengan ilmu pengetahuan dan teknologi yang terus berkembang, penemuan-penemuan dalam bidang fotografi mengawali berkembangnya dunia animasi ke tahap yang lebih kompleks. Pada masa Aristoteles, ia menemukan prinsip-prinsip dasar fotografi melalui reaksi perjalanan cahaya melalui celah sempit. Prinsip inilah yang digunakan pada teknik fotografi selanjutnya, yaitu pengoperasian celah kamera atau lensa untuk memproyeksikan gambar pada kamera. Teknologi tersebut terus berkembang sampai pada akhirnya kamera ditemukan. Kamera pertama adalah sebuah kotak tertutup dengan sebuah lubang kecil pada salah satu dindingnya (*pinhole camera*). Alat ini belum menggunakan lensa, hanya menggunakan lubang sebesar jarum untuk memasukan cahaya, sehingga menghasilkan gambar berupa bayangan walaupun terbalik (Fernandes, 2002).

**2.4 Game**

Game adalah kata berbahasa Inggris yang berarti permainan atau pertandingan, atau bisa diartikan sebagai aktifitas terstruktur yang dilakukan untuk bersenang-senang. Teori permainan adalah suatu cara belajar yang digunakan dalam menganalisa sejumlah pemain maupun perorangan yang menunjukan strategi-strategi rasional. Teori ini dikemukakan oleh John Von Ann dan Oscar Morgenstern. Menurutnya permainan terdiri atas sekumpulan peraturan yang membangun situasi bersaing dari dua atau beberapa orang kelompok dengan memilih strategi yang dibangun untuk memaksimalkan kemenangan sendiri ataupun untuk meminimalkan kemenangan lawan.

Menurut Chris Crawford, seorang designer *game* komputer mengemukakan bahwa game pada intinya adalah sebuah media interaktif, aktifitas yang berpusat pada sebuah pencapaian, ada pelaku aktif, ada lawan anda. "*Games are fundamental part of human existence*".

Game memiliki beberapa jenis, diantaranya adalah *game* shooting (tembak-tembakan), *adventure* (petualangan), *sport* (olahraga), dan *game* edukasi.

**2.5 Game Edukasi**

Menurut Handriyantini, *game* edukasi merupakan permainan yang dirancang untuk memberikan pengajaran pada penggunaannya dari tidak tahu menjadi tahu melalui suatu media yang unik dan menarik. *Game* edukasi ini digunakan untuk memberikan pengajaran atau menambah pengetahuan melalui suatu media yang menarik.

Penerapan *game* edukasi bermula dari perkembangan video *game* yang sangat pesat dan menjadikannya sebagai media efektif yang interaktif dan banyak dikembangkan di perindustrian. Melihat kepopuleran *game* tersebut, para pendidik berpikir bahwa mereka mempunyai kesempatan yang baik untuk menggunakan komponen rancangan *game* dan menerapkannya dalam kurikulum dengan penggunaan industry berbasis *game*. *Game* harus memiliki desain antarmuka yang interaktif dan mengandung unsur menyenangkan. Pengertian *game* computer menurut beberapa ahli :

1. Menurut Chris Crawford, seorang *computer game designer* mengemukakan bahwa *game*, pada intinya adalah sebuah interaktif, aktivitas yang berpusat pada sebuah pencapaian, ada pelaku aktif, ada lawan anda.
2. Menurut David Parlett, *Game* adalah sesuatu yang memiliki "akhir dan cara mencapainya", artinya ada tujuan, hasil dan serangkaian peraturan untuk mencapai keduanya.
3. Menurut Roger Caillois, seorang sosiolog Perancis, dalam bukunya yang berjudul *Les hommes* menyatakan *game* adalah aktivitas yang mencakup karakteristik *fun, separate, uncertain, governed by rules.*

## 2.6 Bahasa Inggris

Bahasa Inggris adalah sebuah bahasa yang berasal dari Inggris, merupakan bahasa utama di Britania Raya, Amerika Serikat, serta banyak Negara lainnya. Bahasa ini berawal dari kombinasi beberapa bahasa lokal yang dipakai oleh orang-orang Norwegia, Denmark dan Angolo-Saxon di abad ke-6 sampai abad ke-10.

Lalu pada tahun 1066 dengan ditaklukan Inggris oleh *William the Conqueror*, sang penakluk dari Normandia, Perancis Utara, maka Bahasa Inggris dengan sangat intensif mulai dipengaruhi Bahasa Latin dan Bahasa Perancis. Dari seluruh kosakata Bahasa Inggris modern, diperkirakan kurang lebih 50% berasal dari Bahasa Perancis dan Latin.

Penggunaan Bahasa Inggris di Indonesia semakin lama semakin umum. Di sekolah, anak-anak sudah mendapatkan pelajaran Bahasa Inggris sejak sekolah dasar atau bahkan sejak taman kanak-kanak. Selain itu semakin banyak sekolah bi-lingual yang menggunakan Bahasa Inggris dan Bahasa Indonesia sebagai bahasa pengantar di sekolah. Di jalan-jalan atau tempat umum, papan reklame atau nama tempat usaha pun banyak yang menggunakan Bahasa Inggris.

Semua hal ini menandakan umumnya penggunaan Bahasa Inggris di Indonesia. Oleh sebab itu sangat penting bagi kita untuk menguasai Bahasa Inggris sebagai bahasa ketiga setelah Bahasa Indonesia dan bahasa daerah tentunya.

## 2.7 Adobe Flash Professional CS6

Adobe Flash Professional CS6 merupakan *tools* yang digunakan dalam pembuatan game edukasi pembelajaran Bahasa Inggris untuk Siswa SD Negeri Cicadas 2. Adobe Flash Professional CS6 memiliki beberapa halaman atau jendela kerja yang mempunyai masing-masing. Diantaranya adalah halaman pembuka atau halaman awal. Halaman awal adalah halaman

atau jendela kerja yang pertama kali muncul ketika mengakses Adobe Flash Professional CS6.

Lingkungan kerja atau *workspace* Adobe Flash Professional CS6 terdiri dari beberapa komponen utama yang terdiri dari :

a. Menu Bar adalah kumpulan yang terdiri atas dasar menu-menu yang digolongkan dalam satu kategori. Misalnya menu *file* terdiri atas perintah *new, open, save, import, export* dan lain-lain.
b. *Timeline* adalah sebuah jendela panel yang digunakan untuk mengelompokkan dan mengatur isi sebuah *movie*, pengaturan tersebut meliputi penentuan masa tayang objek, pengaturan layer dan lain-lain.
c. *Stage* adalah area untuk berkreasi dalam membuat animasi yang digunakan untuk mengkomposisi *frame-frame* secara individual dalam sebuah *movie.*
d. *Toolbox* adalah kumpulan *tools* yang sering digunakan untuk melakukan seleksi, menggambar, mewarnai objek, memodifikasi objek dan mengatur gambar atau objek.
e. *Properties* adalah informasi objek-objek yang ada di *stage*. Tampilan panel properties secara otomatis dapat berganti-ganti dalam menampilkan informasi atribut-atribut *properties* dari objek yang terpilih.
f. *Panel* adalah sebagai pengontrol yang berfungsi untuk mengganti dan memodifikasi berbagai atribut dari objek dari animasi secara cepat dan mudah.

Setelah *workspace* pada Adobe Flash Professional CS6 terdapat juga fasilitas toolbox yang merupakan sekumpulan tools atau alat yang mempunyai fungsi-fungsi tersendiri untuk keperluan desain. Berikut merupakan bagian dari setiap *tools* yang terdapat pada *toolbox* :

a. *Arrow tool* atau sering disebut selection tool berfungsi untuk memilih atau menyeleksi suatu objek.

*b.* *Subselection tool* berfungsi untuk menyeleksi objek lebih detail dari *selection tool.*

c. *Free transform tool* berfungsi untuk mengtransformasi objek yang telah diseleksi.

d. *Lasso tool* berfungsi untuk memotong gambar secara manual.

e. *Pen tool* berfungsi untuk menggambar garis dengan bantuan titik-titik bantu seperti dalam pembuatan garis, kurva atau gambar.

f. *Texttool* berfungsi untuk membuat objek teks.

g. *Line tool* berfungsi untuk membuat garis.

h. *Oval tool* berfungsi untuk membuat lingkaran atau elips.

i. *Pencil* tool berfungsi untuk membuat garis.

j. *Brush tool* berfungsi untuk menggambar bentuk garis-garis dan bentuk-bentuk bebas.

k. *Ink bottle tool* berfungsi untuk mengisi atau mengganti *stroke* (garis luar) suatu objek.

l. *Painbucket tool* berfungsi untuk mengisi area-area kosong atau digunakan untuk mengubah warna area sebuah objek yang telah diwarnai.

m. *Eye dropper tool* berfungsi untuk mengambil stempel warna.

n. *Eraser tool* berfungsi untuk menghapus objek.

o. *Hand tool* berfungsi untuk menggeser tampilan stage tanpa mengubah pembesaran.

p. *Zoom tool* berfungsi untuk merperbesar atau memperkecil tampilan stage.

q. *Stroke colour* berfungsi untuk memilih atau memberi warna pada suatu garis.

r. *Fill colour* berfungsi untuk memilih atau memberi warna pada suatu objek.

s. *Swap colour* berfungsi untuk menukar warna *fill* dan *stroke* atau sebaliknya pada suatu gambar atau objek.

Setelah *toolbox* terdapat pula menu *library*. Fungsi dari *library* adalah sebagai wadah untuk menyimpan program-program terpisah yang sudah jadi, seperti tombol, objek grafis, video, audio dan lain-lain.

## 2.8 *Action Script 3.0*

### 2.8.1 Pengertian

*Action Script* adalah bahasa pemrograman yang dibuat berdasarkan ECMAScript yang digunakan dalam pengembangan situs web dan perangkat lunak menggunakan platform Adobe Flash Player. Bahasa ini awalnya dikembangkan oleh Macromedia, tapi kini sudah dimiliki dan dilanjutkan perkembangannya oleh Adobe, yang membeli Macromedia pada tahun 2005.

Pada Flash, *Action Script* memiliki fungsi dasar, antara lain :

1. *Animation*

   Animasi yang sederhana memang tidak membutuhkan *Action Script*, namun untuk animasi yang kompleks, *Action Script* akan sangat membantu. Sebagai contoh, animasi bola yang memantul di tanah yang mengikuti hokum fisika akan membutuhkan ratusan *frame*. Namun dengan menggunakan *Action Script*, animasi tersebut dapat dibuat hanya dengan satu *frame* saja.

2. Navigasi

   Pergerakan animasi pada Flash secara default bergerak ke depan dari satu *frame* ke *frame* lainnya hingga selesai. Namun dengan *Action Script*, jalannya animasi dapat dikontrol untuk berhenti di suati *frame* dan berpindah ke sembarang *frame* sesuai dengan pilihan dari *user*.

3. *User Input*

   *Action Script* dapat digunakan untuk menerima suatu masukan dari *user* yang kemudian informasi tersebut dikirimkan kepada server untuk diolah. Dengan kemampuan ini, *Action Script* dapat digunakan untuk membangun suatu aplikasi berbasis Flash.

4. Memperoleh Data

Seperti yang telah dijelaskan sebelumnya, *Action Script* dapat melakukan interaksi dengan server. Dengan demikian kita dapat mengupdate informasi lalu menampilkannya kepada *user*.

5. Kalkulasi

Action Script dapat melakukan kalkulasi, misalnya seperti yang diterapkan pada aplikasi *shopping chart.*

6. Grafik

*Action Script* dapat mengubah ukuran sebuah grafik, sudut rotasi, warna movie clip dalam movie, serta dapat menduplikasi dan menghapus item dari *screen*.

7. Mengenali *Environtment*

*Action Script* dapat mengambil nilai waktu dari sistem yang digunakan oleh *user*.

8. Memutar Musik

Selain animasi yang berupa gerakan, pada program Flash juga dapat diinputkan sebuah music sehingga animasi yang dihasilkan menjadi lebih menarik. Pada hal ini *Action Script* dapat digunakan untuk mengontrol *balance* dan volume dari musik tersebut.

### 2.8.2 Penggunaan *Action Script*

Penggunaan *Action Script* adalah untuk mempermudah pembangunan suatu aplikasi atai animasi. Semakin kompleks animasi pada Flash, maka akan semakin banyak memakan *frame*. Dengan *Action Script*, penggunaan *frame* tersebut dapat dikurangi, bahkan dapat membuat animasi yang kompleks hanya dengan satu *frame* saja.

1. Struktur

Flash menggunakan struktur bahasa Dot Syntax. Dot atau titik (.) digunakan untuk menunjukan metoda atau properti yang terkait dengan objek.

2. Objek

Objek adalah suatu tipe data seperti suara, gambar, teks yang digunakan untuk mengontrol movie. Semua objek merupakan bagian dari satu kelas. Objek pada Flash dapat berupa gambar yang Nampak, hingga sesuatu yang tidak Nampak, misalnya tanggal, data atau deteksi input dari mouse. Objek dapat dikenali dan digunakan setelah terlebih dahulu diberi nama. Proses penamaan suatu objek disebut instantiating. Selain objek yang didefinisikan, pada Flash terdapat *predefined class* yang terdiri dari objek yang bias dipakai di dalam movie. Beberapa diantaranya. *Movie Clip, Color, Sound, Button, Stage, Text, Field,* dan *Text Format*.

3. *Class* dan *Function*

Pada dasarnya *Action Script* adalah sebuah bahasa permrograman yang dibangun dari *class-class* yang telah dibuat oleh *developer* Flash. *Programmer* dapat menggunakan *class* tersebut dengan *script-script* yang sederhana dan mudah untuk diingat.

Berikut ini adalah struktur penulisa *class :*

**Class NamaClass {**

**Class body**

**}**

Pada *Action Script* 3.0, *class* didefinisikan pada file *Action Scriptexternal* berekstensi *. Terdapat 2 tipe *class* pada *Action Script*, yaitu :

1. *Built-in Class*, yaitu *class* yang sudah disediakan oleh Flash. *Programmer* dapat memanggilnya dalam bentuk statement yang sederhana.
2. *Custom Class*, yaitu *class* yang dibuat sendiri dengan maksud dan tujuan yang sesuai dengan kebutuhan *programmer* dalam membangun suatu aplikasi.

*Function*/fungsi adalah kumpulan *script* yang mengerjakan operasi untuk tujuan tertentu. Terdapat beberapa jenis yang dapat digunakan dalam membangun suatu aplikasi menggunakan *Action Script* 3.0, yaitu :

1. *Built-in functions*
2. *Named dan user-definied functions*
3. *Anonymous function*
4. *Callback functions*
5. *Constructor function*
6. *Function literal*

### 2.8.3 Mekanisme Kontrol pada Objek dengan *AS 3.0*

Mekanisme pengendalian (*control mechanism*) diwujudkan dengan menggunakan umpan balik (*feedback*), yang mencuplik keluaran. Umpan balik ini digunakan untuk mengendalikan baik masukan maupun proses. Tujuannya adalah untuk mengatur agar sistem berjalan sesuai dengan tujuan.

### 2.8.4 Pengaturan Properties Objek dengan *AS 3.0*

1.Penggunaan perintah *Trace*

Perintah *Trace* digunakan untuk menampilkan informasi pada panel *Output*. Panel *Output* hanya digunakan untuk menguji jalannya program saja, bukan sebagai tampilan utama program. Yang menjadi tampilan utama program adalah *stage* kita.

2. Pedeklarasian Variabel

Variabel-variabel yang akan digunakan dalam *Script* kita harus dideklarasikan terlebih dahulu. Beberapa tipe variabel yang sering digunakan antara lain *TextField, Number, String, Array, Sprite, Shape* dan MovieClip. Kita juga dapat membuat tipe variabel sendiri berupa objek *Class* untuk aplikasi kita. Perintah deklarasi variabel mempunyai bentuk umum :

**var namaVariabel:TipeVariabel;**

atau cara kedua berilut juga memberikan hasil yang sama :

**var namaVariabel:TipeVariabel=newTipeVariabel();**

3. *Event, Listener dan Even Handler*

Semua interaksi *user* di dalam Flash dikelola oleh *"event-event"* beberapa contoh *event* adalah *mouse click, mouse move, input keyboard, loading file,* dan lain-lain. Dalam buku ini akan banyak menggunakan pengaturan *event* untuk mendesain interaksi *user*. Dalam *Action Script,* suatu objkek bias mendengarkan (*listen*) *event* yang terjadi pada dirinya dengan menggunakan *method* addEventListener pada objek tersebut. *Method* ini membutuhkan dua argument, yaitu :

a. *Event* apa yang akan didengarkan, misalnya *MouseEvent* ataupun *KeyboardEvent*
b. Fungsi apa yang akan saya kerjakan apabila saya mendengarkan *event*.
   Contoh :
   tombolMulai.addEventListener(MouseEvent.CLICK,tekantombol);
   function tekanTombol(mevt:MouseEvent):void{ISI FUNGSI}

Mempunyai maksud sebagai berikut :

a. Tombol dengan nama *istance* tombolMulai akan mendengarkan *event Mouse Click* (MouseEvent.CLICK)
b. Apabila tombolMulai di *click* dengan *mouse*, maka fungsi tekanTombol akan dipanggil dan dikerjakan. Fungsi tekanTombol dinamakan Event Handler. Sebuah *event* handler harus berupa fungsi dengan satu nilai *input* (misalnya dalam contoh kita adalah mevt:MouseEvent) dan tidak mengembalikan nilai apa-apa (yaitu mempunyai tipe keluaran vold).

## 2.9 *Speech Recognition*

*Speech recognition* adalah fasilitas *Windows 7* yang digunakan untuk mengendalikan komputer dengan perintah suara (Andi, 2010). Sebelum menggunakan fasilitas ini, pastikan komputer telah terpasang sebuah *microphone. Microsoft speech recognition* memungkinkan pengguna untuk mengontrol komputer dengan perintah suara seperti membuka program, klik tombol, klik link pada *web* dan juga dapat mendikte teks sehingga lebih cepat ditulis daripada mengetik. Untuk melakukan semua itu sebuah komputer membutuhkan sebuah *microphone* sebagai media untuk menghantarkan suara ke komputer. Keakuratan suara yang diterima komputer tergantung pada kualitas *dari microphone* yang digunakan. *Microsoft speech recognition* tersedia dalam Bahasa Inggris, Bahasa Spanyol, Bahasa Prancis, Bahasa Jepang dan Bahasa Cina (Pogue, 2010).

Terdapat dua macam mode pada sistem pengenalan pembicaraan, yaitu :

a. Mode Diktasi

Ketika *user* mengucapkan kata atau kalimat yang selanjutnya akan dikenali oleh komputer dan diterjemahkan ke dalam teks.

b. *Mode Command and Control*

Ketika *user* mengucapkan kata yang sudah terdefinisi yang selanjutnya digunakan untuk menjalankan perintah tertentu.

Penggunaan *Microsoft Speech Recognition* dalam *Windows 7* akan membatu pengguna dalam meningkatkan produktivitas dengan menghemat banyak waktu dalam melakukan perintah tertentu.

## 2.10 *OOP (Object Oriented Programming)*

*OOP (Object Oriented Programming)* atau yang dikenal dengan permrograman berorientasi objek merupakan *paradigm* pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam *paradigm* ini dibungkus ke dalam kelas-kelas atau objek-objek.

Model data berorientasi objek dikatakan dapat memberi fleksibelitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung *OOP* mengklaim bahwa *OOP* lebih mudah dipelajari bagi pemula disbanding dengan pendekatan sebelumnya, dan pendekatan *OOP* lebih mudah dikembangkan dan dirawat.

Dengan menggunakan *OOP* maka dalam melakukan pemecahan suatu masalah kita tidak melihat bagaimana cara menyelesaikan suatu masalah tersebut (terstruktur) tetapi objek-objek apa yang dapat melakukan pemecahan masalah tersebut. Sebagai contoh anggap kita memiliki sebuah departemen yang memiliki manager, sekretaris, petugas administrasi data dan lainnya. Missal manager tersebut ingin memperoleh data dari bagian administrasi maka manager tersebut tidak harus mengambilnya langsung tetapi dapat menyuruh petugas bagian administrasi untuk mengambilnya. Pada kasus tersebut seorang manager tidak harus mengetahui bagaimana cara mengambil data tersebut tetapi manager bias mendapatkan data tersebut melalui objek petugas administrasi. Jadi untuk menyelesaikan suatu masalah dengan kolaborasi antar objek-objek yang ada karena setiap objek memiliki deskripsi tugasnya sendiri.

## 2.11 *Unified Modelling Language (UML)*

*UML* singkatan dari *Unified Modeling Language* yang berarti bahasa pemodelan standar. (Chonoles, 2003:bab1) mengatakan sebagai bahasa, berarti *UML* memiliki sintaks dan semantik. Ketika membuat model menggunakan konsep *UML* ada aturan-aturan yang harus diikuti. Bagaimana elemen pada model-model yang dibuat berhubungan satu dengan yang lainnya harus mengikuti standar yang ada. *UML* bukan hanya sekedar diagram, tetapi juga meceritakan konteksnya.




**Gambar 2.2 Logo *UML***

Jenis-jenis diagram dalam *UML* yang akan digunakan diantaranya sebagai berikut :

a. Diagram *Use-Case (Use Case Diagram)*
b. Diagram Kelas *(Class Diagram)*
c. Diagram *Sequence* (*Sequence Diagram*)
d. Diagram Aktivitas (*Activity Diagram*)

### 2.11.1 *Use-Case Diagram*

*Use case diagram* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah actor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

*Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang test *case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain

sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali use case yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *berhaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

### 2.11.2 *Activity Diagram*

*Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses *parallel* yang mungkin terjadi pada beberapa eksekusi.

*Activity diagram* merupakan state diagram khusus, dimana sebagian besar state adalah action dan sebagian besar transisi di-*trigger* oleh selesainya state sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan *behavuiour internal* sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari *level* atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana actor menggunakan sistem untuk melakukan aktivitas.

Sama seperti *state*, standar *UML* menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behavior* pada kondisi tertentu. Untuk mengilustrasikan proses-proses parallel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal.

### 2.11.3 *Class Diagram*

*Class diagram* adalah sebuah soesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class diagram* menggambarkan keadaan atribut atau *property* suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan metode atau fungsi.

*Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi dan lain-lain. *Class* memiliki tiga area pokok, yaitu berupa nama, atribut dan metoda.

### 2.11.4 *Sequence Diagram*

*Sequence* diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence* diagram terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

*Sequence diagram* biasa digunakan untuk menggambarkan scenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan.

Masing-masing objek termasuk actor memiliki lifeline *vertical. Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya message akan dipetakan menjadi operasi atau metoda dari class. *Activation bar* menunjukan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message.*

## 2.12 Metode Pengujian Sistem

Metode pengujian sistem untuk mengetahui efektifitas dari *software* yang digunakan selain memberikan kesempatan kepada pengguna untuk mengoperasikan dan melakukan pengecekan terhadap laporan yang dihasilkan melalui *software.*

### 2.12.1 Pengujian *Black-box*

Pengujian black-box adalah pengujian yang dilakukan untuk antarmuka perangkat lunak. Pengujian ini dilakukan untuk memperlihatkan bahwa fungsi-fungsi bekerja dengan baik dalam arti masukan yang diterima dengan benar dan keluaran yang dihasilkan benar-benar tepat dan pengintegrasian dari eksternal data berjalan dengan baik.

Metode pengujian *black-box* ini memfokuskan pada *requirement* fungsi dari perangkat lunak, pengujian ini merupakan komplementari dari pengujian *white-box.* Pengujian *white-box* dilakukan terlebih dahulu pada proses pengujian, sedangkan pengujian *black-box* dilakukan pada tahap akhir dari pengujian perangkat lunak.

Proses yang terdapat dalam proses pengujian *black-box* yaitu analisis batasan nilai yang berlaku untuk setiap data.

### 2.12.2 Skala Likert

Skala Likert digunakan untuk mengukur sikap, pendapat dan persepsi seorang atau kelompok orang tentang fenomena social. Dalam penelitian, fenomena social ini telah ditetapkan secara spesifik oleh peneliti, yang selanjutnya disebut sebagai variabel penelitian.

Contoh :

Preferensi

1. Sangat Tidak Setuju
2. Tidak Setuju
3. Cukup Setuju

4. Setuju

5. Sangat Setuju

Untuk keperluan analisis kuantitatif, maka jawaban tersebut diberi nilai skor, yaitu :

**Tabel 2.2 Skala Likert**

| Jawaban | Skor |
|---|---|
| Sangat Tidak Setuju | 1 |
| Tidak Setuju | 2 |
| Cukup Setuju | 3 |
| Setuju | 4 |
| Sangat Setuju | 5 |

Skala Likert menujukan kesepakatan atau ketidaksepakatan dengan setiap pernyataan dalam instrument. Setiap respon diberi skor *numeric*, menunjukan *favourableness* atau *unfavourableness,* dan skor dijumlahkan untuk mengukur sikap responden. Dengan kata lain, nilai keseluruhan merupakan responden di posisi kontinum *favourableness* atau *unfavourableness* terhadap masalah.