BAB II
LANDASAN TEORI

2.1 Simulasi

Menurut Hasan (2002), simulasi merupakan suatu model pengambilan keputusan dengan mencontoh atau mempergunakan gambaran sebenarnya dari suatu sistem kehidupan dunia nyata tanpa harus mengalaminya pada keadaan yang sesungguhnya.

Pada umumnya terdapat 5 langkah pokok yang diperlukan dalam menggunakan simulasi, yaitu :

1. Menentukan persoalan atau sistem yang hendak disimulasi.
2. Formulasikan model simulasi yang hendak digunakan.
3. Ujilah model dan bandingkan tingkah lakunya dengan tingkah laku dari sistem nyata, kemudian berlakukanlah model simulasi tersebut.
4. Rancang percobaan – percobaan simulasi.
5. Jalankan simulasi dan analisis data (Levin, dkk, 2002).

Beberapa kelebihan model simulasi adalah sebagai berikut :

1. Simulasi memberikan solusi jika model analitik gagal melakukannya.
2. Model simulasi lebih realistis terhadap sistem nyata karena memerlukan asumsi yang lebih sedikit.
3. Perubahan konfigurasi dan struktur dapat dilaksanakan lebih mudah untuk menjawab pertanyaan : *what happen if.*
4. Dalam banyak hal, simulasi lebih murah dari percobaannya sendiri.
5. Simulasi dapat digunakan untuk maksud pendidikan.

Namun, model simulasi juga memiliki beberapa kekurangan, yaitu :

1. Simulasi bukanlah presisi dan juga bukan suatu proses optimisasi. Simulasi tidak menghasilkan solusi, tetapi ia menghasilkan cara untuk menilai solusi termasuk solusi optimal.

https://lib.mercubuana.ac.id

2. Model simulasi yang baik dan efektif sangat mahal dan membutuhkan waktu yang lama dibandingkan dengan model analitik.
3. Tidak semua situasi dapat melalui simulasi kecuali situasi yang memuat ketidakpastian.

2.2 Konsep Dasar Kompresi Data

Proses kompresi data didasarkan pada kenyataan bahwa pada hampir semua jenis data selalu terdapat pengulangan pada komponen data yang dimilikinya, misalnya didalam suatu teks kalimat akan terdapat pengulangan penggunaan huruf. Kompresi data melalui proses *encoding* berusaha untuk menghilangkan unsur pengulangan ini dengan mengubahnya sedemikian rupa sehingga ukuran data menjadi lebih kecil, baik untuk berbagai jenis data seperti data teks, gambar, video, suara, dan lain-lain.

Proses pengurangan unsur pengulangan ini dapat dilakukan dengan memakai beberapa teknik kompresi. Misalnya jika suatu komponen muncul berulang kali dalam suatu data, maka komponen tersebut tidak harus dikodekan berulang kali pula tapi dapat dikodekan dengan menulis frekuensi munculnya komponen dan di mana komponen tersebut muncul. Teknik kompresi data lainnya, berusaha untuk mencari suatu bentuk kode yang lebih pendek untuk suatu komponen yang sering muncul.

Data tidak hanya disajikan dalam bentuk audio (bunyi, suara, musik) maupun video, tetapi juga dapat berupa teks dan citra. Keempat macam data tersebut sering disebut dengan multimedia. Pada umumnya representasi data digital membutuhkan memori yang besar, disisi lain kebanyakan data misalnya citra (image) mengandung duplikasi. Duplikasi ini dapat berarti dua hal yaitu pertama, besar kemungkinan suatu pixel dengan pixel lain tetangganya memiliki intensitas yang sama, sehingga penyimpanan setiap pixel memboroskan tempat.

Kedua, citra banyak mengandung bagian (region) yang sama, sehingga bagian yang sama ini tidak perlu dikodekan berulang kali. Saat ini, kebanyakan aplikasi menginginkan representasi dengan memori yang lebih sedikit. Pemampatan data atau kompresi data (data compression) bertujuan meminimalkan kebutuhan memori untuk merepresentasikan data digital. Prinsip umum yang

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

digunakan pada proses kompresi adalah mengurangi duplikasi data sehingga memori untuk merepresentasikan menjadi lebih sedikit daripada representasi data digital semula .

Data digital yang telah dikompresi dapat dikembalikan ke bentuk data digital semula (dekompresi) dimana hal ini tergantung pada aplikasi software yang mendukung kompresi tersebut. Ketika suatu aplikasi mampu 'menghilangkan' atau mengkompresi data yang tidak dibutuhkan maka aplikasi tersebut juga mampu mengembalikan data yang dihilangkan tersebut sehingga menjadi data digital semula (dekompresi) namun terdapat juga suatu aplikasi yang dapat mengkompresi namun ketika dekompresi dapat menggunakan aplikasi lain contohnya aplikasi winzip dengan aplikasi winrar.

2.2.1 Kompresi Lossless

Pada teknik ini tidak ada kehilangan informasi. Jika data dimampatkan secara *lossless*, data asli dapat direkontruksi kembali sama persis dari data yang telah dimampatkan, dengan kata lain data asli tetap sama sebelum dan sesudah pemampatan. Secara umum teknik *lossless* digunakan untuk penerapan yang tidak bisa mentoleransi setiap perbedaan antara data asli dan data yang telah direkonstruksi. Data berbentuk tulisan misalnya *file* teks, harus dimampatkan menggunakan teknik *lossless*, karena kehilangan sebuah karakter saja dapat mengakibatkan kesalapahaman. *Lossless compression* disebut juga dengan *reversible compression* karena data asli bisa dikembalikan dengan sempurna. Akan tetapi rasio kompresinya sangat rendah, misalnya pada data teks, gambar seperti GIF dan PNG. Contoh metode ini adalah *Shannon-Fano coding*, *Huffman coding*, *Arithmetic coding* dan *Run-length encoding*.

2.2.2 Kompresi Lossy

Pada teknik ini akan terjadi kehilangan sebagian informasi. Data yang telah dimampatkan dengan teknik ini secara umum tidak bisa direkonstruksi sama persis dari data aslinya. Di dalam banyak penerapan, rekonstruksi yang tepat bukan suatu masalah. Sebagai contoh, ketika sebuah sampel suara ditransmisikan, nilai eksak dari setiap sample suara belum tentu diperlukan. Tergantung pada

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

yang memerlukan kualitas suara yang direkonstruksi, sehingga banyaknya jumlah informasi yang hilang di sekitar nilai dari setiap sample dapat ditoleransi.

Biasanya teknik ini membuang bagian-bagian data yang sebenarnya tidak begitu berguna, tidak begitu dirasakan, tidak begitu dilihat sehingga manusia masih beranggapan bahwa data tersebut masih bisa digunakan walaupun sudah dikompresi. Misalnya pada gambar dan MP3. Contoh metode ini adalah *Transform Coding*, *Wavelet*, dan lain-lain. *Lossy compression* disebut juga *irreversible compression* karena data asli mustahil untuk dikembalikan seperti semula. Kelebihan teknik ini adalah rasio kompresi yang tinggi dibanding metode *lossless*.

2.2.3 Teori Umum Kompresi Data

Beberapa teknik pengompresan data secara *lossless* yang dapat digunakan pada berbagai tipe data yaitu:

1. Run Length Encoding

   Run Length Encoding adalah suatu teknik kompresi yang menggantikan simbol yang berurutan dengan simbol itu sendiri diikuti dengan jumlah perulangan yang terjadi. Sebagai contoh, *string* 111110000003355 dapat diekspresikan dengan 15063252. Sudah jelas bahwa teknik kompresi ini sangat berguna untuk simbol-simbol yang berurutan dan berulang. Dengan demikian, teknik kompresi ini berguna untuk *file-file* gambar dimana pixel-pixel gambar tersebut mempunyai nilai yang sama.

2. Huffman Coding

   Huffman Coding yaitu teknik pengompresan data yang menempatkan *Variable Length Codes*(VLC) ke dalam simbol-simbol, sehingga simbol-simbol yang sering muncul memiliki kode yang paling singkat. Pada proses dekompresi simbol-simbol tersebut akan dikembalikan ukuran kode aslinya. Ketika digunakan untuk mengompres teks, contohnya *Variable Length Codes* digunakan sebagai ganti kode-kode ASCII, dan karakter-karakter yang sering muncul, biasanya spasi,c dan a digunakan sebagai kode-kode terpendek. Dengan cara ini jumlah total bit yang diperlukan untuk mentransmisikan data dapat dipertimbangkan lebih sedikit dibandingkan jika kode yang diambil

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

sama untuk semua karakter. Huffman Coding secara khusus lebih efektif jika simbol pada data tidak banyak variasi.

3. Arithmetic Coding

   Walaupun Huffman Coding sangat efisien, teknik tersebut hanya optimal jika kemungkinan simbol-simbol adalah 2^n. Arithmetic Coding tidak mempunyai pengecualian ini dan biasanya lebih efisien dibandingkan dengan teknik Huffman Coding yang lebih popular. Walaupun lebih efisien, algoritma Arithmetic Coding lebih rumit.

4. Lempel-Ziv Coding

   Lempel-Ziv coding menggunakan sebuah kamus simbol-simbol yang berurutan. Ketika sebuah urutan tersebut diulang maka akan diganti dengan suatu referensi ke posisinya dalam kamus tersebut. Ada beberapa variasi dari teknik Lempel-Ziv coding dan masing-masing berbeda dalam pengaturan kamusnya. Teknik kompresi data dari Lempel-Ziv yang paling terkenal yaitu Lempel-Ziv Welch.

2.3 Teori Run Length Encoding

Berbeda dengan teknik-teknik sebelumnya yang bekerja berdasarkan karakter per karakter, teknik run length ini bekerja berdasarkan sederetan karakter yang berurutan. *Run Length Encoding* adalah suatu algoritma kompresi data yang bersifat lossless. Algoritma ini mungkin merupakan algoritma yang mudah untuk dipahami dan diterapkan untuk kompresi. *Run Length Encoding* (*RLE*) adalah algoritma kompresi yang sangat mendasar. Metode kompresi ini sangat sederhana, yaitu hanya memindahkan pengulangan byte yang sama berturut-turut (secara terus menerus).

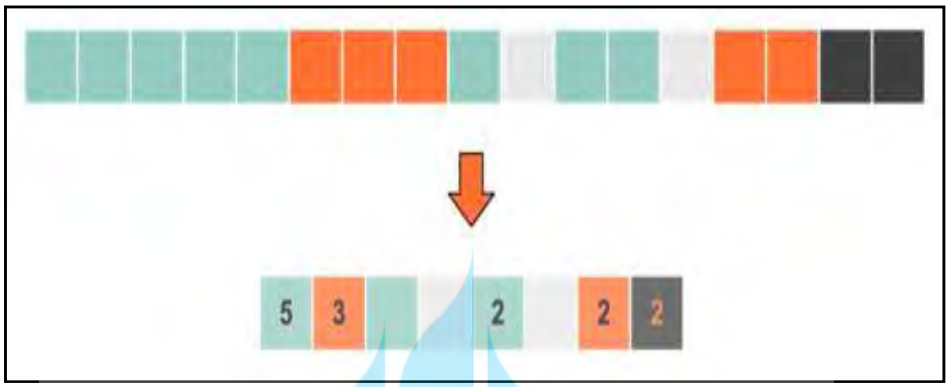
Teknik yang digunakan pada algoritma ini adalah dengan mencari karakter yang berulang lebih dari 3 kali pada suatu file untuk kemudian diubah menjadi sebuah bit penanda (*marker bit*) diikuti oleh sebuah bit yang memberikan informasi jumlah karakter yang berulang dan kemudian ditutup dengan karakter yang dikompres. Sayangnya teknik ini tidak berguna pada file-file dokumen umum dimana kemungkinan penggunaan karakter sangat heterogen. Di sisi lain

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

teknik kompresi ini akan sangat efektif untuk *file-file* gambar, karena gambar memiliki elemen piksel yang berulang dengan warna yang identik .

Pada gambar 2.1 kita dapat memampatkan piksel berturut-turut dengan hanya mengganti masing-masing data yang berjalan dengan satu pixel dan counter yang menunjukkan berapa banyak item di dalamnya.



Gambar 2.1 Contoh Kompresi Algoritma Run Length Encoding

Deretan data bagian atas merupakan deretan data pada file asli, sedangkan deretan data bagian bawah merupakan deretan data hasil pemampatan dengan algoritma *Run Length Encoding*. Dalam hal ini kita hanya dapat menyimpan counter untuk piksel yang diulang lebih dari sekali . Seperti input stream " aaaabbaba " akan dikompresi sebagai " [4] a [2] baba ".

UNIVERSITAS MERCU BUANA

2.3.1 Kompresi dan Dekompresi

Pada dasarnya semua data itu merupakan rangkaian bit 0 dan 1 yang membedakan antara suatu data tertentu dengan data yang lain adalah ukuran dari rangkaian bit dan bagaimana 0 dan 1 itu ditempatkan dalam rangkaian bit tersebut. Misalnya data berupa audio, dalam data audio suatu rangkaian bit tertentu mewakili satu nada. Semakin kompleks suatu data, ukuran rangkaian bit yang diperlukan semakin panjang, dengan demikian ukuran keseluruhan data juga semakin besar.

Dalam penyimpanan dan pengiriman data (transmisi), selain isi dari data tersebut parameter yang tidak kalah pentingnya adalah ukurannya. Kerap kali data yang disimpan dalam suatu media penyimpanan berukuran sangat besar sehingga

https://lib.mercubuana.ac.id

memerlukan tempat yang lebih banyak dan tidak efisien. Apalagi bila data tersebut akan dikirim, semakin besar ukurannya, waktu yang diperlukan untuk pengiriman akan lebih lama. Untuk itu, diperlukan kompresi data (data compression) untuk memperkecil ukuran suatu data tanpa merubah isi atau informasi yang terkandung dalam data tersebut.

Kompresi data merupakan suatu upaya untuk mengurangi jumlah bit yang digunakan untuk menyimpan atau mentransmisikan data. Kompresi data meliputi berbagai teknik kompresi yang diterapkan dalam bentuk perangkat lunak (*software*) maupun perangkat keras (*hardware*). Bila ditinjau dari sisi penggunaannya, kompresi data bisa bersifat umum untuk segala keperluan atau bersifat khusus untuk keperluan tertentu.

Keuntungan data yang terkompresi antara lain:

1. mengurangi *bottleneck* pada proses I/O dan transmisi data
2. penyimpanan data lebih hemat ruang
3. mempersulit pembacaan data oleh pihak yang tidak berkepentingan
4. memudahkan distribusi data dengan media removable seperti flash disk, CD, DVD, dll.

Data digital yang telah dikompresi dapat dikembalikan ke bentuk data digital semula (dekompresi) dimana hal ini tergantung pada aplikasi *software* yang mendukung kompresi tersebut. Ketika suatu aplikasi mampu 'menghilangkan' atau mengkompresi data yang tidak dibutuhkan maka aplikasi tersebut juga mampu mengembalikan data yang dihilangkan tersebut sehingga menjadi data digital semula (dekompresi) namun terdapat juga suatu aplikasi yang dapat mengkompresi namun ketika dekompresi dapat menggunakan aplikasi lain contohnya aplikasi winzip dengan aplikasi winrar.

Dalam *Run-Length Encoding,* langkah-langkah kompresi adalah sebagai berikut:

1. Baca *file input* dari data yang akan dikompresi.
2. Lihat apakah terdapat deretan karakter yang sama secara berurutan lebih dari tiga karakter, jika terdapat hal tersebut lakukan kompresi.
3. Berikan bit penanda pada *file* kompresi, bit penanda ini berfungsi untuk menandai bahwa karakter selanjutnya adalah karakter yang dikompresi

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

sehingga tidak membingungkan pada saat mengembalikan *file* yang sudah dikompresi ke *file* aslinya.

4. Tambahkan deretan bit untuk menyatakan jumlah karakter yang sama berurutan.
5. Tambahkan deretan bit yang menyatakan karakter yang berulang.

Untuk melakukan proses dekompresi dilakukan langkah-langkah sebagai berikut:

1. Baca *file* hasil kompresi data.
2. Lihat karakter pada hasil kompresi satu-persatu dari awal sampai ahir, jika ditemukan bit penanda, lakukan proses pengembalian.
3. Lihat karakter setelah bit penanda, konversikan ke bilangan decimal untuk menentukan jumlah karakter yang berurutan.
4. Lihat karakter berikutnya, kemudian lakukan penulisan karakter tersebut sebanyak bilangan yang telah diperoleh pada karakter sebelumnya (langkah 3).

Dalam mengimplementasikan program kompresi menggunakan algoritma ini, perlu diperhatikan jika pada *file* asli terdapat karakter yang sama dengan bit penanda, maka pada *file encoding* karakter tersebut ditulis sebanyak dua kali secara berurutan. Pada saat pengembalian ke *file* asli, jika ditemukan bit penanda yang berderetan sebanyak dua kali, hal itu berarti karakter tersebut bukan bit penanda, tapi karakter asli dari *file* aslinya.

UNIVERSITAS MERCU BUANA

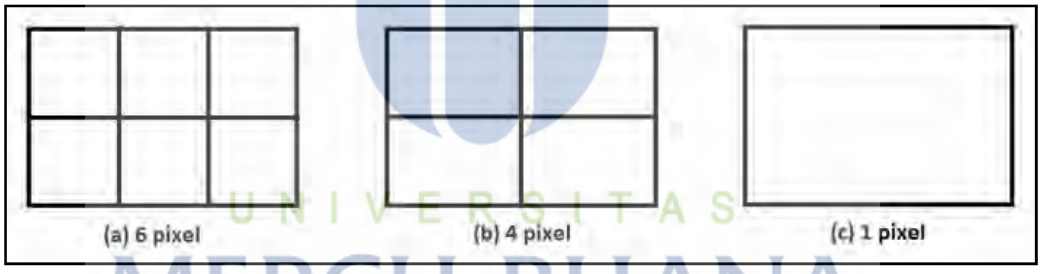
2.4 Piksel

Piksel adalah unsur gambar atau representasi sebuah titik terkecil dalam sebuah gambar grafis yang dihitung per inci. Piksel sendiri berasal dari akronim bahasa Inggris *Picture Element* yang disingkat menjadi *Pixel*. Pada ujung tertinggi skala resolusi, mesin cetak gambar berwarna dapat menghasilkan hasil cetak yang memiliki lebih dari 2.500 titik per inci denga pilihan 16 juta warna lebih untuk setiap inci, dalam istilah komputer berarti gambar seluas satu inci persegi yang bisa ditampilkan pada tingkat resolusi tersebut sepadan dengan 150 juta bit informasi.

https://lib.mercubuana.ac.id

Monitor atau layar datar yang sering kita temui terdiri dari ribuan piksel yang terbagi dalam baris-baris dan kolom-kolom. Jumlah piksel yang terdapat dalam sebuah monitor dapat kita ketahui dari resolusinya. Resolusi maksimum yang disediakan oleh monitor adalah 1024x768, maka jumlah pixel yang ada dalam layar monitor tersebut adalah 786432 piksel. Semakin tinggi jumlah piksel yang tersedia dalam monitor, semakin tajam gambar yang mampu ditampilkan oleh monitor tersebut.

Bentuk informasi yang paling mudah diterima manusia adalah dalam bentuk gambar/citra/*image*. Citra berarti gambaran dari objek sebenarnya. Ada dua macam citra, yakni citra analog dan citra digital. Komputer hanya mampu mengolah data dalam bentuk digital, sehingga seluruh citra harus diubah ke dalam bentuk citra digital sebelum diproses oleh komputer. Informasi visual yang menyusun sebuah citra direpresentasikan oleh komputer digital sebagai kumpulan titik yang tersusun dalam matriks. Setiap titik disebut *pixel* (*picture element*) dan setiap pixel dipetakan oleh satu atau lebih bit dalam memori komputer.



Gambar 2.2 Ilustrasi Pixel

Pixel memiliki kaitan erat dengan resolusi. Resolusi adalah perkalian elemen matriks yang menyusun gambar. Dalam gambar ilustrasi, ketiganya merupakan gambar dengan bentuk dan ukuran yang sama, namun dengan jumlah pixel dan resolusi yang berbeda.

Pada gambar 2.2 (a) merupakan gambar dengan resolusi 3x2, dengan jumlah 6 pixel, gambar 2.2 (b) merupakan gambar yang memiliki resolusi 2x2, dengan jumlah 4 pixel dan gambar 2.2 (c) merupakan gambar yang memiliki resolusi 1x1, dengan jumlah 1 pixel.

https://lib.mercubuana.ac.id

Pixel dan resolusi bersama-sama menentukan kualitas gambar. Semakin besar jumlah pixel dan resolusinya, maka semakin baik gambar tersebut. Hal ini dapat dilihat pada foto digital, misalnya. Foto yang diambil dengan kamera 1,3 Megapixel tentu akan kurang baik jika dicetak dalam kertas berukuran A3, lain halnya jika diambil dengan kamera 8 Megapixel.

Semakin banyak titik-titik pixel, maka semakin besar memori yang diperlukan untuk menyimpan informasi gambar tersebut, semakin baik pula kualitas gambar yang dihasilkan.

Salah satu program yang dapat digunakan untuk menghitung jumlah pixel suatu citra secara langsung adalah Matrox Inspector 2.1, yang juga mampu memberikan kepada *user* histogram komponen warna (dalam RGB atau *red green blue*), histogram tingkat keabuan citra, serta menghitung jumlah pixel dalam sebentuk citra yang diinginkan. Untuk mengetahui jumlah pixel tidak hanya terbatas menggunakan Matrox Inspector 2.1 saja, kita dapat membuat sebuah program khusus dengan menggunakan Delphi atau MATLAB, yang tentu saja output keluarannya harus sinergis dengan *software* yang sudah diakui secara internasional.

2.5 Citra Digital

Sebuah citra digital adalah kumpulan piksel-piksel yang disusun dalam larik atau matriks dua dimensi. Indeks baris dan kolom (x,y) dari sebuah piksel yang dinyatakan dalam bilangan bulat dan nilai-nilai tersebut mendefinisikan suatu ukuran intensitas cahaya pada titik tersebut. Satuan atau bagian terkecil dari suatu citra disebut piksel (picture element).

Umumnya citra dibentuk dari persegi empat yang teratur sehingga jarak horizontal dan vertikal antara piksel satu dengan yang lain adalah sama pada seluruh bagian citra. Piksel (0,0) terletak pada sudut kiri atas pada citra, dimana indeks x bergerak ke kanan dan indeks y bergerak ke bawah. Untuk menunjukkan koordinat (m-1,n-1) digunakan posisi kanan bawah dalam citra berukuran m x n pixel.

Citra digital direpresentasikan dalam matriks berukuran m x n sesuai dengan ukuran m x n ukuran citra digital dalam pixel. Jadi setiap pixel dari citra

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

digital diwaklili oleh sebuah matriks berukuran 1x1 apabila citra tersebut berupa citra Grayscale atau citra Biner, dan 3 matriks apabila citra digital berupa citra RGB. Misalkan kita punya citra digital berukuran 640x480 pixel, maka sebenarnya citra tersebut diwakili oleh matriks berukuran 640x480.

2.6 Adobe Flash CS6

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 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 fiture-fiture 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

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

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.

2.6.1 Bahasa Pemrograman Actiont Script 3.0

Actionscript 3.0 atau disingkat AS3 adalah bahasa pemograman yang terdapat pada flash setelah release flash versi ke 9 yang telah di akusisi oleh adobe sehingga namanya menjadi Adobe Flash CS3. AS3 ini merupakan pengambangan dari versi sebelumnya yang dirasa kurang cepat dalam performa dan lambat dalam pengolahan data.

AS3 kini telah dibundle untuk beberapa produk milik adobe lainnya seperti Adobe Flash Builder dan Adobe Flex. Dari segi bahasa, AS3 ini tidak jauh berbeda dengan versi sebelumnya. Hanya ada beberapa perubahan struktur bahasa yang lebih dekat dengan bahasa pemograman Java.

2.7 Unified Modelling Language (UML)

Unified Modelling Language yang sering dikenal dengan istilah UML merupakan sebuah bahasa yang telah menjadi standar dalam bidang visualisasi, perancangan dan dokumentasi sistem perangkat lunak. UML merupakan bagian terpenting dalam pengembangan perangkat lunak yang berorientasi objek. Dengan menggunakan UML kita dapat membuat model semua jenis aplikasi, perangkat lunak, dimana aplikasi tersebut dapat berjalan pada perangkat keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun.

UML mendefinisikan notasi dan semantik (*syntax*). Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram perangkat lunak. Semantik UML mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan.

2.7.1 Diagram Use Case

Diagram use case (*use case diagram*) menggambarkan interaksi antara seorang pengguna sistem dengan sebuah sistem. Dua komponen utama dalam

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

diagram use case adalah use case dan aktor. Aktor merepresentasikan entitas manusia atau mesin yang akan berinteraksi dengan sistem yang akan dirancang. Sedangkan use case merupakan kegiatan atau fungsi yang dilakukan oleh aktor. Use case berisi "apa" yang dilakukan oleh sistem / apa yang terjadi dengan sistem, bukan "bagaimana" sistem melakukannya. Komponen-komponen dalam diagram usa case dapat dilihat pada tabel 2.1.

Diagram use case sangat bermanfaat bagi kita ketika sedang membuat persyaratan sebuah sistem, mempresentasikan rancangan kita dengan klien, dan merancang kasus pengujian untuk semua fitur yang ada pada sistem. Sebuah use case dapat memasukkan (*include*) use case lainnya sebagai bagian dari proses dalam dirinya. Secara umum use case yang dimasukkan akan dipanggil setiap kali use case yang memasukkannya dieksekusi secara normal. Selain itu, use case juga dapat memperluas (*extend*) use case lain sesuai dengan perilakunya sendiri.

Di dalam use case terdapat asosiasi yang digunakan untuk menggambarkan bagaimana aktor terlibat di dalam use case. Ada dua jenis asosiasi yaitu :

1. Include adalah jenis asosiasi yang digunakan untuk menggambarkan pemanggilan use case oleh use case lainnya, contoh : pemanggilan fungsi program.
2. Extend merupakan perluasan dari use case lain jika kondisi atau syarat terpenuhi.

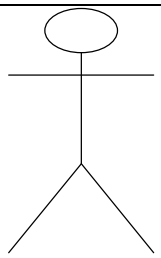
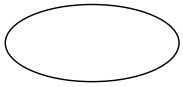
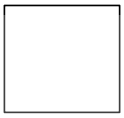
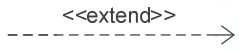
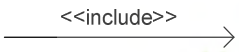
UNIVERSITAS MERCU BUANA

Langkah-langkah dalam membuat diagram use case antara lain :

1. Identifikasi semua aktor,
2. Identifikasi semua use case,
3. Urutkan prioritas use case,
4. Perincian untuk setiap use case,
5. Identifikasi adanya generalisasi tiap use case,
6. Identifikasi hubungan include,
7. Identifikasi hubungan extend.

https://lib.mercubuana.ac.id

Tabel 2.1 Notasi Diagram Use Case (Rosa A.S-Shalahuddin, 2011)

Notasi	Nama	Keterangan
	Aktor	Aktor adalah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. Aktor diberi nama sesuai dengan tugasnya ketika menggunakan sistem. Jangan memberi nama aktor dengan nama orang. Aktor dapat juga berupa sistem lain di luar sistem yang dikembangkan.
	Use case	*Use case* adalah kegiatan atau fungsi yang dilakukan oleh aktor, diberi nama sesuai fungsi yang dilakukan.
	Sistem	Sistem berupa ruang lingkup aplikasi yang dikembangkan.
	Hubungan (*Relationship*)	Hubungan yang dilakukan oleh aktor dengan use case yang dilakukan oleh aktor tersebut.
	Ekstensi (*extend*)	Relasi tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan itu: biasanya use case tambahan memiliki nama depan sama dengan use case yang ditambahkan.
	*include*	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini.

UNIVERSITAS MERCU BUANA

2.7.2 Diagram Aktivitas

Diagram aktivitas (*Activity diagram*) menggambarkan berbagai aliran aktivitas dalam sebuah sistem yang sedang dirancang, bagaimana masing-masing alir berawal, keputusan yang mungkin terjadi, dan bagaimana semua aktivitas itu berakhir. Diagram aktivitas juga dapat menggambarkan proses pararel yang mungkin terjadi pada beberapa eksekusi.

Diagram aktivitas merupakan tahapan (*state*) khusus pada diagram, di mana sebagian besar tahapan adalah aksi (*action*) dan sebagian besar transisi dipicu oleh selesainya status sebelumnya (pemrosesan internal atau internal processing). Oleh karena itu diagram aktivitas tidak menggambarkan tingkah laku

https://lib.mercubuana.ac.id

internal sebuah sistem secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas secara umum. Sebuah aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. Notasi diagram aktivitas ditunjukkan pada tabel 2.2.

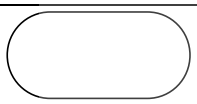
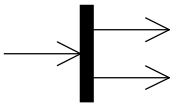
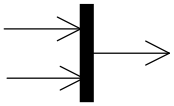
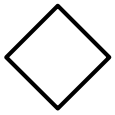
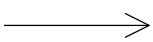
Beberapa alasan membuat diagram aktivitas adalah sebagai berikut :

1. Menganalisis diagram use case lebih rinci,
2. Mengidentifikasi kondisi sebelum dan sesudah suatu use case,
3. Menemukan use case baru yang tersembunyi.

Langkah-langkah dalam membuat diagram aktivitas antara lain :

1. Identifikasi use case,
2. Pemodelan awal untuk setiap use case,
3. Menambahkan garis batas (*swimlane*),
4. Menyaring aktivitas level akhir ke dalam diagram aktivitas.

Tabel 2.2 Notasi Diagram Aktivitas (Rosa A.S-Shalahuddin, 2011)

Notasi	Nama	Keterangan
	Tahapan aksi (*Action state*)	Kegiatan atau proses yang dilakukan.
	Titik mulai (*start point*)	Terminasi awal proses dilakukan.
	Titik akhir (*End point*)	Terminasi akhir proses yang dilakukan.
	Fork (*Percabangan*)	Percabangan proses pada kondisi tertentu.
	Join (*Penggabungan*)	Penggabungan proses, dari kondisi percabangan sebelumnya.
	Decision (*Keputusan*)	Digunakan untuk menggambarkan tingkah laku (*behaviour*) pada kondisi tertentu.
	Pengatur aliran (*Control flow*)	Cara untuk mengelompokkan aktivitas berdasar aktor (mengelompokkan aktivitas pada urutan yang sama).

UNIVERSITAS MERCU BUANA

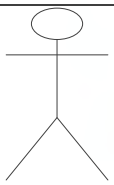
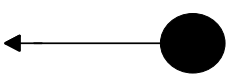
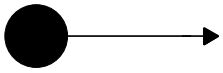
https://lib.mercubuana.ac.id

2.7.3 Diagram Sekuen

Diagram sekuen (*sequence diagram*) menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah kejadian (*event*) untuk menghasilkan keluaran tertentu. Panah digambarkan dari satu objek ke objek lain.

Diagram sekuen digunakan juga untuk menggambarkan beberapa objek dilengkapi dengan pesan yang dikirim atau diterima oleh setiap objek. Oleh karena itu, untuk menggambar diagram sekuen harus diketahui objek-objek yang terlibat dalam sebuah *use case*. Banyaknya diagram sekuen yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri. Notasi diagram sekuen ditunjukkan pada tabel 2.3.

Tabel 2.3 Notasi Diagram Sekuen (Rosa A.S-Shalahuddin, 2011)

No	Notasi	Keterangan
1.		Aktor, yang digunakan untuk menggambarkan pelaku atau pengguna dalam *use case*, pelaku ini meliputi manusia atau sistem komputer atau subsistem lain yang memiliki metode untuk melakukan sesuatu.
2.		Objek, digunakan untuk menggambarkan pelaku atau pengguna dalam diagram *sequence*. Pelaku ini meliputi manusia atau sistem komputer atau subsistem lain yang memiliki metode untuk melakukan sesuatu.
3.		*Lifeline,* digunakan untuk mempresentasikan sebuah individu dalam interaksi dan hanya sebuah entitas interaksi.
4.		*ExecutionSpecification,* digunakan untuk menggambarkan spekifikasi dari sebuah unit kelakuan atau aksi antar *lifeline.*
5.	message	Pesan (*message*), digunakan untuk mendeskripsikan pesan yang ada antar *lifeline.*
6.		*Lost message*, digunakan untuk menggambarkansebuah pesan yang mendefinisikan komunikasi particular antara lifelines dalam interaksi dari *lifeline* ke *lifeline*
7.		*Found message*, digunakan untuk menggambarkan sebuah pesan yang mendefinisikan komunikasi particular antara lifelines dalam interaksi *lifeline* ke *lifeline*

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

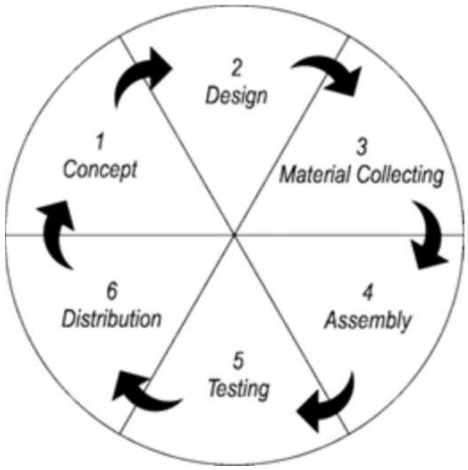
2.8 Storyboard

Storyboard digunakan sebagai alat bantu pada tahapan perancangan multimedia. Storyboard merupakan pengorganisasian grafik, contohnya adalah sederetan ilustrasi atau gambar yang ditampilkan berurutan untuk keperluan visualisasi awal dari suatu file, animasi, atau urutan media interaktif, termasuk interaktivitas di web. Keuntungan menggunakan storyboard adalah sebagai pedoman dari aliran pekerjaan yang akan dilakukan oleh pembuat multimedia, (Binanto, 2010).

Storyboard dapat juga dikatakan sebagai *visual script* yang akan dijadikan outline dari sebuah proyek, ditampilkan rekaman demi rekaman yang biasa disebut dengan istilah *scene*. Sebuah storyboard media interaktif dapat digunakan dalam antarmuka grafik pengguna untuk rancangan rencana desain sebuah *website* atau proyek interaktif sebagaimana alat visual untuk perancangan ini.

2.9 Metode Pengembangan Aplikasi (LUTHER)

Banyak metodologi Pengembangan Perangkat Lunak (*Software Engineering*), tetapi tidak pas diterapkan pada pengembangan perangkat lunak berbasis Multimedia. Begitu pula pada penelitian ini, metode yang dipakai dalam aplikasi simulasi kompresi data ini. Salah satunya adalah menurut Sutopo (2003), yang berpendapat bahwa metodologi Pengembangan multimedia terdiri dari 6 tahapan, yaitu *concept, design, material collecting, assembly, testing* dan *distribution* seperti gambar di bawah ini:



Gambar 2.3 Metode Pengembangan Aplikasi

https://lib.mercubuana.ac.id

1. Konsep (*Concept*)

   Tahap *concept* (pengonsepan) adalah tahap untuk menentukan tujuan dan siapa pengguna program (identifikasi *audiens*). Selain itu menentukan jenis aplikasi (presentasi, interaktif, dan lain – lain) dan tujuan aplikasi (hiburan, pelatihan, pembelajaran, dan lain – lain).

2. Perancangan (*Design*)

   Perancangan adalah tahap pembuatan spesifikasi mengenai arsitektur program, tampilan, dan kebutuhan material / bahan untuk program.

3. Pengumpulan Bahan (*Material Collecting*)

   Pengumpulan bahan adalah tahap dimana pengumpulan bahan yang sesuai dengan kebutuhan yang dilakukan seperti teks, gambar, animasi, suara, dan lain – lain yang diperlukan untuk tahap berikutnya.

4. Pembuatan (*Assembly*)

   Pembuatan simulasi kompresi data adalah tahap dimana semua objek yang dibutuhkan dibuat. Pembuatan aplikasi didasarkan pada tahap perancangan yang di lanjutkan ke tahap pembuatan apa yang telah direncanakan pada tahap perancangan.

5. Pengujian (*Testing*)

   Tahap pengujian dilakukan setelah menyelesaikan tahap pembuatan dengan menjalankan aplikasi dan melihatnya apakah tepat sasaran pada tujuan pembuatan aplikasi tersebut serta adanya kesalahan atau tidak. Fungsi dari pengujian ini adalah untuk melihat apakah hasil pembuatan aplikasi sesuai dengan yang direncanakan.

6. Distribusi

   Tahap ini dilakukan setelah tahap pengujian selesai, maka diadakannya penulisan sistem yang bertujuan sebagai instruksi atau buku petunjuk. Pada tahap ini disusun buku sebagai dokumentasi dari pelaksanaan tugas akhir.

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id