PERANCANGAN DAN ANALISIS KOMPRESI *AUDIO* WAV DENGAN MENGGUNAKAN METODE *HUFFMAN*

Eko Surahman

Program Studi Teknik Informatika

STMIK Indonesia Mandiri, Jl.Jakarta No.79 Bandung

Email : ekosurahman01@gmail.com

ABSTRAK

Permasalahan yang dihadapi oleh pengguna teknologi informasi salah satunya adalah ukuran *file* yang besar sehingga diperlukan suatu media penyimpanan yang besar serta diperlukan waktu yang cukup lama untuk menyimpannya. Solusi untuk masalah diatas adalah dengan melakukan pemapatan/ kompresi. Ada banyak sekali metode kompresi yang ada saat ini, salah satunya adalah metode kompresi *Huffman*. Metode ini menggunakan prinsip dengan memperkecil bit yang sering muncul dan memperbesar bit yang jarang muncul. Kompresi dilakukan dengan cara membuat pohon biner *Huffman.* Analisis metode algoritma *Huffman* ini bertujuan untuk membuat aplikasi yang dapat mengkompresi *file audio* WAV dengan menggunakan metode *Huffman*. Serta untuk mengetahui hasil *file audio* WAV setelah dikompresi. Dalam pengembangannya aplikasi kompresi ini menggunakan metode *waterfall* dimana ia merupakan metode yang umum digunakan dalam pengembangan *software*. Dari hasil pengujian proses kompresi didapat bahwa rasio mempunyai besaran antara 65 % untuk nilai terkecil dan 94% untuk nilai tertinggi. Jika dicari hasil rasio kompresi tersebut secara rata-rata adalah sebesar 82,85%. Ini berarti ukuran *file* hasil adalah 0,8285 kali ukuran *file*. Dengan rata-rata pengurangan *file* sekitar 17,15%. *Audio* WAV yang telah di kompresi dengan algoritma *Huffman* ini dapat di kembalikan lagi dengan cara melakukan dekompresi.

Kata Kunci *: Audio, Huffman,* Kompresi,Dekompresi, WAV, *Waterfall*.

1. PENDAHULUAN

Pesatnya perkembangan teknologi informasi dan komunikasi pada saat ini sangat memungkinkan manusia untuk melakukan pengiriman data dan informasi dengan cepat. Informasi tidak hanya disajikan dalam bentuk teks saja, melainkan data dan informasi dapat juga berupa gambar, suara dan *video* yang lebih dikenal dengan multimedia. *File audio* WAV merupakan salah satu *format file* suara yang banyak dipakai dalam sistem operasi *Windows* untuk keperluan *game* dan multimedia. *File audio* WAV sebenarnya merupakan *format* kasar

(*raw format*) dimana *signal* suara langsung direkam dan dikuantisasi menjadi data *digital*. *Format* dasar dari *file* ini secara *default* tidak mendukung kompresi dan dikenal dengan nama PCM (*Pulse Code Modulation*).

Semakin lama durasi sebuah *file audio* WAV, semakin besar kapasitas media penyimpanan yang dibutuhkan untuk menyimpan data *audio file* tersebut. Media penyimpanan yang semakin besar tidak akan menjawab kebutuhan teknologi informasi jika data berkas (*file*) yang digunakan juga semakin besar. Masalah tersebut dapat diatasi bila *file audio* WAV tersebut dikompresi untuk mengurangi ukurannya. Salah satunya adalah dengan menggunakan metode kompresi *Huffman*.

Metode *Huffman* adalah salah satu metode kompresi yang paling terkenal untuk mengkompres data. Terdapat tiga *fase* untuk menggunakan metode *Huffman* untuk mengkompres sebuah data, pertama adalah *fase* pembentukan pohon *Huffman*, kedua adalah *fase encoding* (kompresi) dan ketiga *fase decoding* (dekompresi) . Prinsip kerja metode ini adalah dengan memperkecil bit yang sering muncul dan memperbesar bit yang jarang muncul. Sebagai contoh, data yang awalnya memiliki 56 bit, bisa di kompresi menjadi 11 bit tanpa ada informasi yang hilang.

Berdasarkan masalah di atas, penulis tertarik membuat penelitian dengan judul "*PERANCANGAN DAN ANALISIS KOMPRESI AUDIO WAV DENGAN MENGGUNAKAN METODE HUFFMAN*".

1.1. Identifikaasi Masalah

Dapat disimpulkan bahwa berdasarkan latar belakang yang telah di uraikan diatas, identifikasi masalah yang terdapat pada tugas akhir ini adalah :

1. Bagaimana cara merancang aplikasi kompresi *file audio* WAV dengan menggunakan metode *Huffman*?
2. Bagaimana hasil *file audio* WAV setelah di kompresi?

1.2. Tujuan Penelitian

Berdasarkan rumusan masalah diatas, tujuan penelitian ini adalah :

1. Merancang aplikasi yang dapat mengkompresi *file audio* WAV dengan menggunakan metode *Huffman*.
2. Untuk mengetahui hasil *file audio* WAV setelah di kompresi.

2. LANDASAN TEORI

2.1. Kompresi Data

Kompresi Data adalah proses pemadatan data yang bertujuan untuk memperkecil ukuran data sehingga selain dapat menghemat media penyimpanan data dalam jaringan (Howe,1993). Kompresi data juga didefinisikan sebagai suatu proses mengubah suatu input data menjadi data lain dengan *format* dan dengan ukuran yang lebih kecil (Solomon, 2004).

2.2. Dekompresi Data

Kebalikan dari proses kompresi data yaitu proses dekompresi. Dekompresi adalah sebuah proses untuk mengembalikan data baru yang telah dihasilkan oleh proses kompresi menjadi data awal. Dekompresi yang menghasilkan data sama persis dengan data aslinya sebelum kompresi, maka data tersebut disebut *lossless compression*. Sebaliknya, jika hasil dekompresi menghasilkan data tidak sama persis dengan data aslinya sebelum dekompresi, karena ada data yang dihilangkan karena dirasa tidak terlalu penting tetapi tidak mengubah informasi yang dikandungnya, disebut *lossy compression*. Setelah dilakukan proses kompresi terhadap sebuah *file*, maka sebuah *file* akan dapat dikembalikan ke dalam bentuk semula yaitu dengan melakukan dekompresi (*decompression*) pada data *file* tersebut (Kharisma, 2010).

2.3. Algoritma

Dalam matematika dan komputasi algoritma merupakan kumpulan perintah yang saling berkaitan untuk menyelesaikan suatu masalah. Perintah-perintah ini dapat diterjemahkan secara bertahap dari awal hingga akhir. Dalam penyusunannya diperlukan urutan serta logika agar algoritma yang dihasilkan sesuai dengan yang diharapkan. Algoritma merupakan bagian terpenting yang tidak dapat dipisahkan dari pemograman. Meskipun sintaksis dan semantik yang dibuat benar adanya, dengan algoritma yang keliru, permasalahan yang ingin dipecahkan dengan teknik pemograman tidak akan berhasil. Oleh karena itu, sebelum membuat *program* aplikasi, hal pertama yang harus kita pahami algoritma atau prosedur pemecahannya.

Hal ini bertujuan agar *program* yang telah dibuat dapat sesuai dengan yang diharapkan (Ramadhani, 2015).

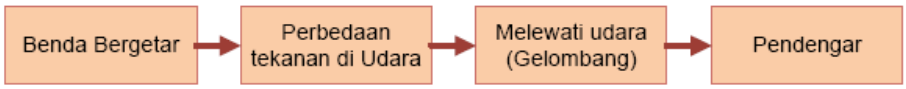
2.4. Algoritma *Huffman*

Algoritma *Huffman* ditemukan oleh *Huffman* pada tahun 1952 dan dipublikasikan pada *paper* yang berjudul *" A method for the construction of minimum-redudancy codes"* Algoritman *Huffman* dapat dipakai sebagai algoritma pencarian (*Searching*) dan algoritma pemapat data (*Data Compression*) (Widyawardhana, 2000).

Algoritma *Huffman* dapat diterapkan untuk melakukan pencarian karena pohon *Huffman* di bentuk dengan meletakkan *item* yang paling sering muncul di daun yang paling dekat dengan akar dan *item* yang paling jarang muncul di tempatkan pada daun yang paling jauh dengan akar. Pencarian dilakukan dengan membaca pohon tersebut satu persatu mulai dari akar, sehingga dalam proses pencarian *item* yang sering muncul dapat ditemukan. Dengan asumsi bahwa pencarian akan sering diperlukan untuk *item* yang sering muncul maka algoritma ini menjadi efektif.

2.5. *Audio*

Audio (Suara) adalah fenomena fisik yang dihasilkan oleh getaran suatu benda yang berupa sinyal *analog* dengan *amplitude* yang berubah secara kontiniu terhadap satuan waktu yang disebut frekuensi. Selama bergetar, perbedaan tekanan terjadi di udara sekitarnya. Pola osilasi yang terjadi dinamakan sebagai gelombang. Gelombang mempunyai pola sama yang berulang pada *interval* tertentu, yang disebut sebagai periode. Contoh suara periodik adalah instrumen musik, nyanyian burung sedangkan contoh suara non periodik adalah batuk, percikan ombak dan lain-lain (Binanto, 2010).



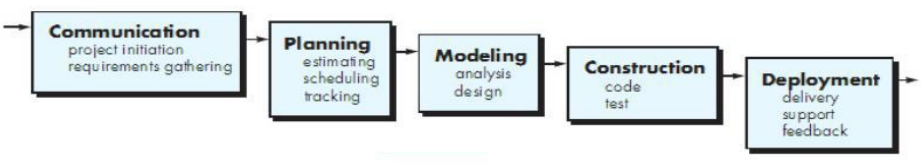
GAMBAR : 2.1. Alur gelombang suara (Binanto, 2010)

2.6. *File* WAV

WAV adalah format *audio* standar *Microsoft* dan IBM untuk personal *computer* (PC), biasanya menggunakan *coding* PCM (*Pulse Code Modulation*). WAV adalah data tidak terkompres sehingga seluruh sampel *audio* disimpan semuanya di *harddisk*. *Software* yang dapat menciptakan WAV dari *analog sound* misalnya adalah *Windows Sound Recorder*. *File audio* ini jarang sekali digunakan di internet karena ukurannya yang relatif besar dengan batasan maksimal untuk *file* WAV adalah 2 GB (Meckah, 2005).

2.7. Metode Perancangan

Metode perancangan yang digunakan dalam tugas akhir ini adalah menggunakan salah satu dari metode pengembangan perangkat lunak *software development life cycle* (SDLC) yaitu metode *waterfall* yang memiliki beberapa tahapan yaitu :



GAMBAR : 2.2. Model *Waterfall* (Pressman, 2015 : 42)

2.8. *Flowchart*

Flowchart merupakan penggambaran secara grafik dari langkah-langkah dan urutan prosedur suatu *program*, Biasanya mempengaruhi penyelesaian masalah yang khusunya perlu dipelajari dan dievaluasi lebih lanjut.(Indrajani, 2011:22).

2.9. UML (*Unified Modelling Language*)

Unified Modeling Language (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem (Windu, 2013 : 4).

3. ANALISI MASALAH DAN PERANCANGAN PROGRAM

3.1. *Communication*

Pada tahap ini dilakukan proses pengumpulan informasi berupa data yang berkaitan dengan penelitian.

3.1.1. Pengumpulan Data

Pengumpulan data merupakan suatu hal yang penting dalam penelitian, karena ini merupakan stategi ataupun cara yang dipakai oleh peneliti untuk mengumpulkan data yang berkaitan dengan kompresi *audio* berekstensi WAV guna mendukung penelitian yang sedang dilakukan.

3.1.1.1. Studi Literatur

Peneliti melakukan studi literatur dengan cara membaca dan mempelajari secara mendalam buku-buku referensi dan jurnal - jurnal penelitian sejenis mengenai kompresi *audio* WAV dan metode *Huffman*.

3.1.2. Analisis Sistem

Kompresi *file* yang akan dibangun mengimplementasikan metode *Huffman*, pengkompresian *file* yang dapat memampatkan ukuran *file* sesuai dengan jenis *file* yang telah ditentukan. Proses yang dapat dilakukan pada sistem yang akan dirancang mencakup kompresi *file* dan dekompresi *file*, Desain dan implementasi rancangan meliputi desain *interface*, gambaran proses sistem, implementasi desain, dan semua yang diperlukan dalam aplikasi kompresi *file* yang akan dibangun. Aplikasi kompresi *file* ini mampu memanpatkan *file* berekstensi WAV.

3.1.3. Analisis Identifikasi Masalah

Analisis masalah merupakan langkah dimana langkah ini diperlukan untuk mengertahui permasalahan yang akan terjadi di dalam sistem yang dirancang oleh penulis. Indentifikasi masalah dalam penelitian ini adalah bagaimana cara merancang aplikasi kompresi *file audio* WAV menggunakan metode *Huffman* serta untuk mengetahui berapa hasil dari *file audio* WAV setelah dilakukan proses kompresi.

3.1.4. Analisis Perangkat Keras

Analisis Kebutuhan perangkat keras bertujuan untuk mengetahui spesifikasi perangkat keras apa saja yang sedang digunakan oleh aplikasi ini diantaranya adalah sebagai berikut.

1. *Processor* AMD A9-9400 (2Cpu) 2.4 GHz.
2. *Memory* (RAM) 4GB.
3. *Harddisk* Minimal 6GB.
4. VGA *card* minimal 128 MB
5. *Mouse*
6. *Keyboard*

3.1.5. Analisis Perangkat Lunak

Perangkat lunak yang digunakan dalam aplikasi ini diantaranya adalah sebagai berikut :

1. Sistem Operasi *Windows* 10
2. *Microsoft Visual Basic 2010.*

3.1.6. Analisis Kompresi Algoritma *Huffman*

Kode *Huffman* merupakan angka-angka biner yang digunakan untuk mengkodekan setiap karakter dalam *file*. Pohon Huffman (*Huffman Tree*) adalah karakter-karakter yang akan direpresentasikan dalam biner, dipisahkan ke dalam cabang pohon biner dan diberi frekuensinya. Cabang sebelah kiri diberi bit 0 dan bit kanan diberi angka 1 sebagai identitas (Fardisa dan Budiono, 2011). Jadi, bit ini akan dibaca dari akar sampai ke daun sehingga terbentuk angka biner sebagai identitas setiap karakter. Adapun langkah-langkah untuk membentuk pohon *Huffman* adalah sebagai berikut :

1. Baca semua data untuk menentukan frekuensi kemunculan masing-masing simbol pada sampel *file audio*.
2. Urutkan simbol sampel *audio* berdasarkan frekuensi kemunculannya secara *Ascending* (terkecil ke yang terbesar).
3. Gabung dua simpul bebas yang mempunyai frekuensi kemunculan paling kecil pada kumpulan simpul. Kemudian dibuat simpul pertama pada pohon *Huffman* dimana simpul tersebut mempunyai frekuensi yang merupakan hasil penjumlahan dari dua simpul penyusunnya.
4. Masukkan simpul pertama ke dalam kumpulan simpul dan urutkan kembali berdasarkan frekuensi kemunculannya, dari yang terkecil ke yang terbesar.

5. Ulangi langkah 2-4 sampai tersisa hanya satu pohon biner. Agar pemilihan dua pohon yang akan digabungkan berlangsung cepat, maka semua pohon yang ada selalu terurut menaik berdasarkan frekuensi.
6. Beri label setiap sisi pada pohon biner dengan cara sisi kiri pohon diberi label 0 dan sisi kanan pohon diberi label 1.
7. Telusuri pohon biner dari akar ke daun. Barisan label-label sisi dari akar kedaun menyatakan kode *Huffman* untuk simbol yang bersesuaian.

3.1.7. Analisis Dekompresi Algoritma *Huffman*

Dekompresi adalah kebalikan dari kompresi yaitu penyusunan kembali sampel audio yang telah dikompresi menjadi sampel semula. Sebelum melakukan dekompresi *file*, maka harus dibentuk kembali pohon *Huffman* berdasarkan informasi yang disimpan dengan *file* yang dikompresi. Proses dekompresi dapat dilakukan dengan dua cara, yang pertama adalah dengan pohon *Huffman* dan yang kedua menggunakan tabel kode *Huffman*. Seperti pada contoh kompresi *Huffman* atau kompresi sebelumnya. Untuk melakukan dekompresi dapat dilakukan dengan pohon *Huffman* sebagai berikut :

1. Proses pembacaan simpul dimulai dari akar pohon *Huffman*, dengan membaca sebuah bit.
2. Untuk setiap bit pada langkah 1, lakukan *traversal* (Kunjungan dalam pohon, dengan setiap *node* hanya dikunjungi tepat satu kali) pada cabang yang bersesuaian.
3. Lakukan pengulangan langkah 1 dan 2 sampai cabang yang bersesuaian ketemu setiap bagian simpul mana sampel itu berada.
4. Lakukan pengulangan langkah 1 sampai seluruh sampel *audio* tersebut di dekompresi.

3.2. *Planning*

Dalam tahap ini penelitian memfokuskan pada penjadwalan pengerjaan penelitian. Pada penelitian ini terdapat beberapa proses yang harus dilakukan dari tahap *communication*, *implementation* sampai dengan *testing* maka dari itu diperlukan penjadwalan yang tepat agar penelitian ini dapat selesai sesuai dengan waktunya.

3.3. *Modelling*

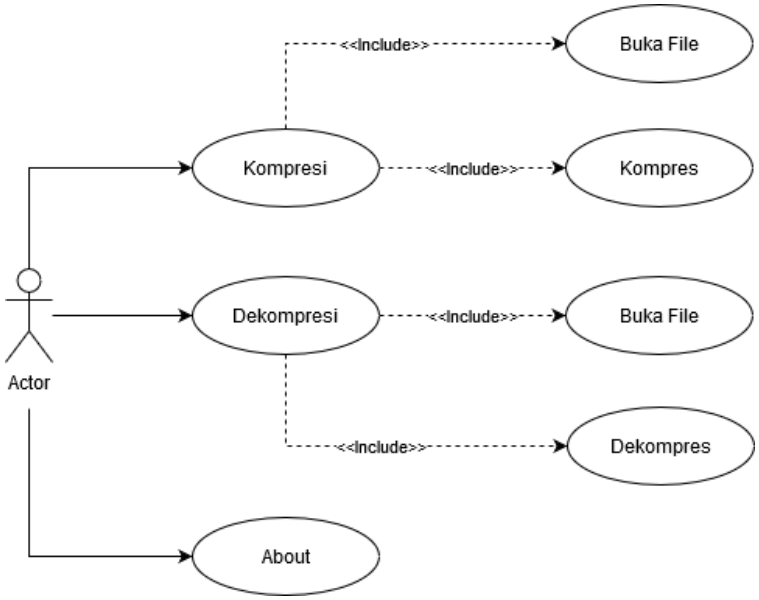
Tahapan ini merupakan proses perancangan aplikasi yang dituangkan kedalam bentuk *use case*, *activity*, *sequence,* dan *flowchart*. Penulis dalam tahap ini membuat *use case*, *activity*, *sequence* dan *flowchart* sesuai dengan tahap yang dibutuhkan.

3.3.1. Perancangan Sistem

Pada tahap ini dilakukan perancangan sistem yang dibutuhkan. Perancangan sistem perlu dilakukan agar memberikan gambaran yang jelas dan lengkap rancangan bangun dan implementasi bagaimana sistem dibuat.

3.3.1.1. *Use case Diagram*

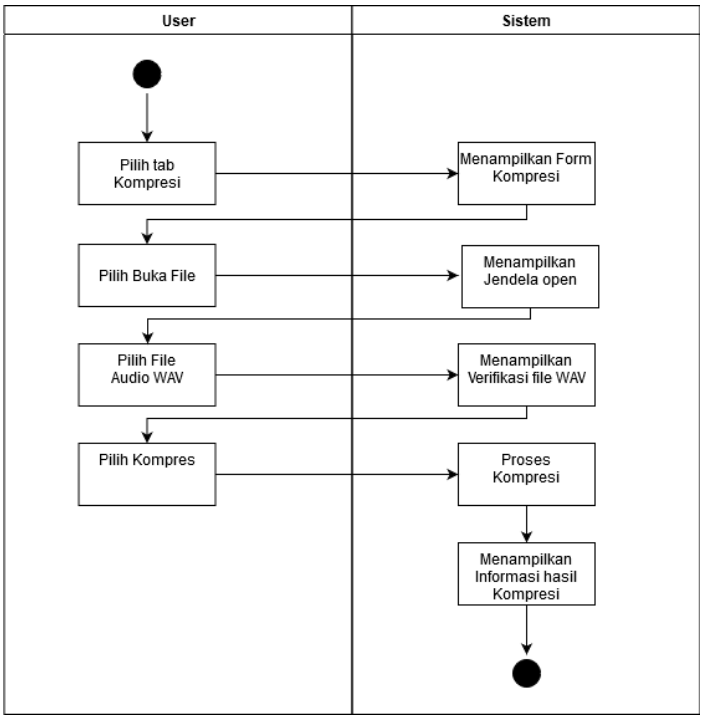
Use-case diagram adalah diagram yang mendeskripsikan interaksi antara *user* dengan sistem. Sebuah *use-case diagram* dapat menggambarkan semua kegiatan didalam satu sistem yang berjalan.



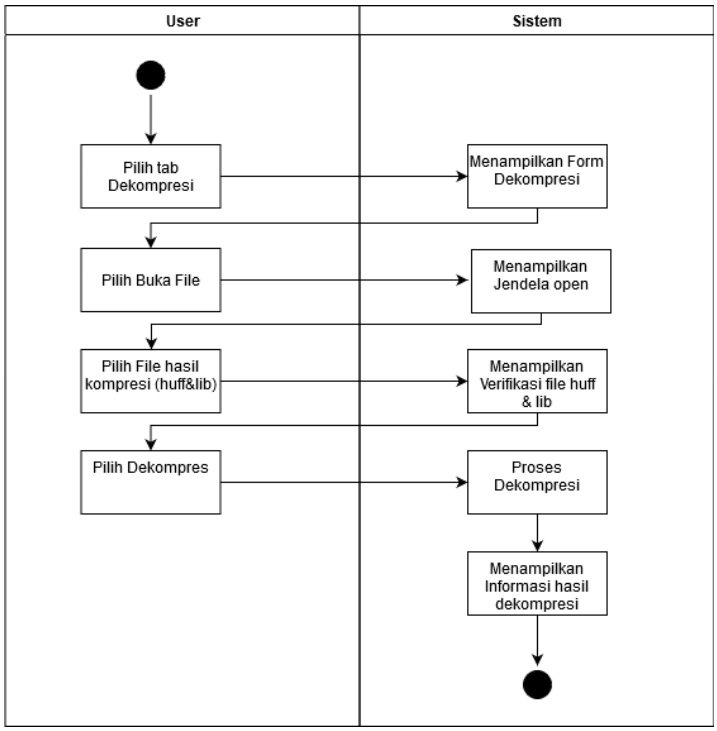
GAMBAR : 3.1. *Use Case Diagram*

3.3.1.2. *Activity Diagram*

Activity diagram adalah suatu diagram yang digunakan untuk menggambarkan aliran proses usaha secara grafis, langkah-langkah dari suatu *use-case*, atau logika dari suatu tingkah laku objek (*method*).



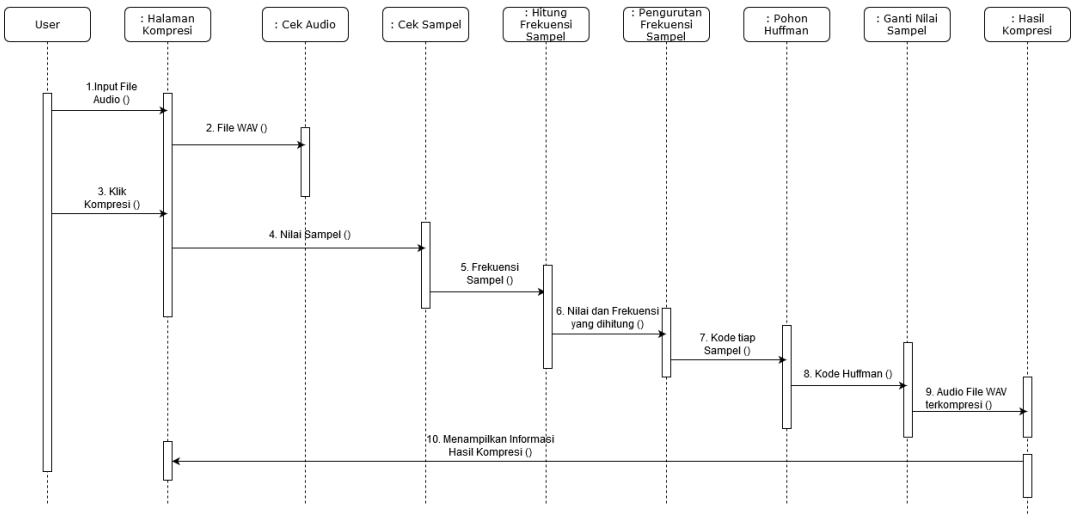
GAMBAR : 3.2. *Activity Diagram* Kompresi



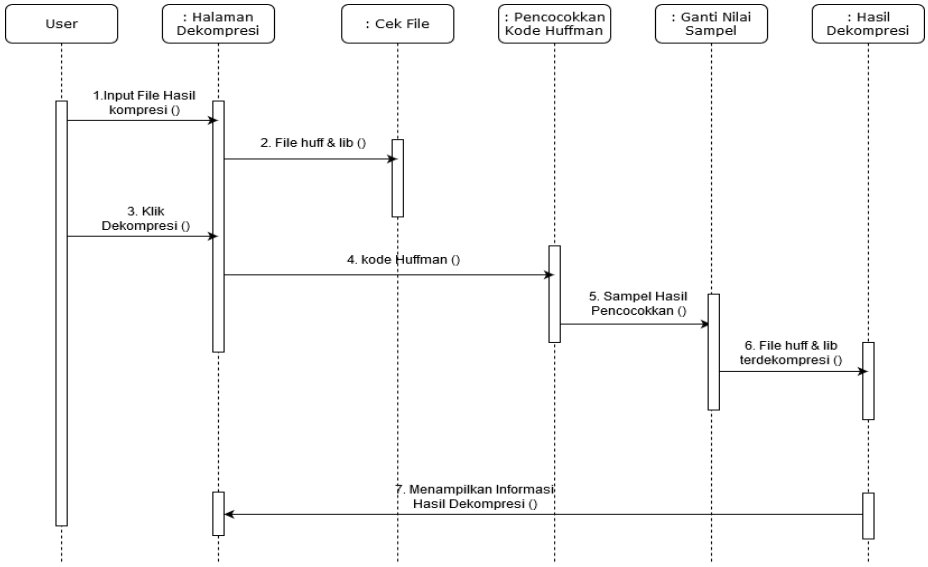
GAMBAR : 3.3. *Activity Diagram* Dekompresi

3.3.1.3. *Sequence Diagram*

Sequence diagram adalah diagram yang memodelkan logika sebuah *use-case* dengan cara menggambarkan interaksi pesan di antara objek – objek dalam rangkaian waktu

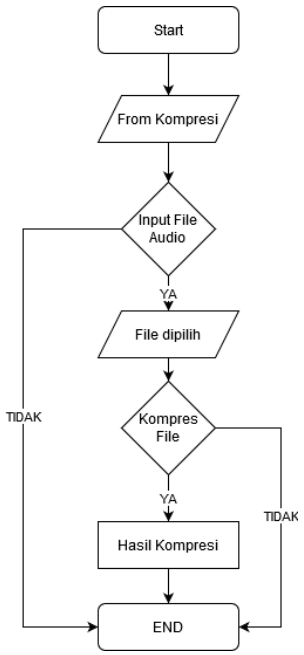


GAMBAR : 3.4. *Sequence Diagram* Kompresi



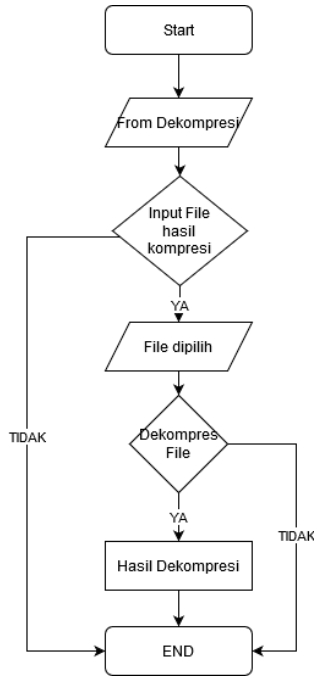
GAMBAR : 3.5. *Sequence Diagram* Dekompresi

3.3.1.4. *Flowchart* kompresi



GAMBAR : 3.6. *Flowchart* Proses Kompresi

3.3.1.5. *Flowchart* Dekompresi



GAMBAR : 3.7. *Flowchart* Proses Dekompresi

4. IMPLEMTASI DAN UJI COBA

4.1. *Construction (Code & Test)*

Pada tahap ini hasil rancangan akan diimplemtasikan menjadi sebuah aplikasi berbasis *desktop* yang dibangun menggunakan bahasa pemograman *VB.Net* 2010, sistem ini memiki dua proses yaitu proses kompresi dan dekompresi.

4.1.1. Implentasi Hardware & Sofware

Implementasi *Hardware & Software* mengacu pada hasil analisis perangkat keras & lunak seperti yang terurai pada Bab III. Untuk lebih jelasnya akan dijelaskan dibawah ini :

4.1.1.1. Perangkat keras (*Hardware*)

Spesifikasi *Hardware* yang digunakan aplikasi ini adalah :

1. *Processor* AMD A9-9400 (2Cpu) 2.4 GHz.
2. *Memory* (RAM) 4GB.

3. *Harddisk* Minimal 6GB.

4.1.1.2. Perangkat Lunak (*Software*)

1. Sistem Operasi *Windows* 10
2. *Microsoft Visual Basic* 2010

4.2. Implementasi Algoritma

Pada aplikasi ini terdiri dari 2 proses utama yaitu kompresi dan dekompresi. Pada proses kompresi dilakukan beberapa tahapan seperti pembacaan jumlah simbol, mengurutkan setiap simbol, pembetukkan pohon *Huffman* dan terakhir adalah menggabungkan 2 *node* pohon. Selanjutnya pada proses dekompresi hanya melakukan pengembalian dan pencarian data yang sebelumnya sudah di simpan pada *library*.

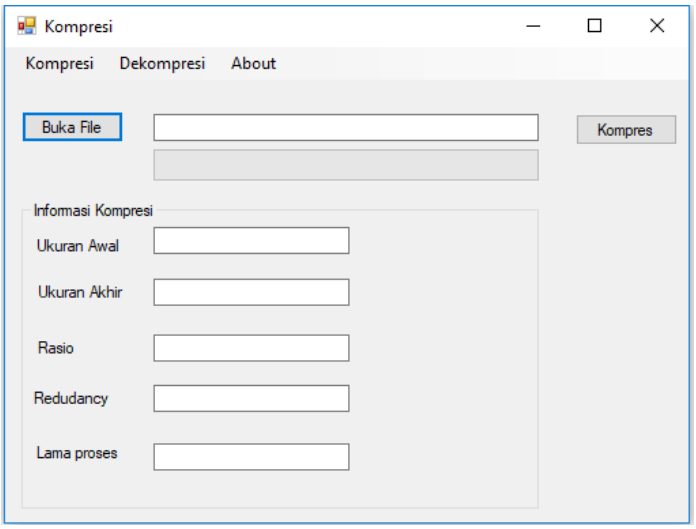
4.3. Implementasi Antar Muka

Implementasi Antar muka merupakan pemaparan mengenai tampilan aplikasi dan kegunaan fungsi dari setiap *Form* yang ada. Untuk memperjelas bentuk dari implementasi antarmuka, berikut pemaparan dan fungsi dari setiap tampilan yang telah dibuat.

1. Tampilan Utama

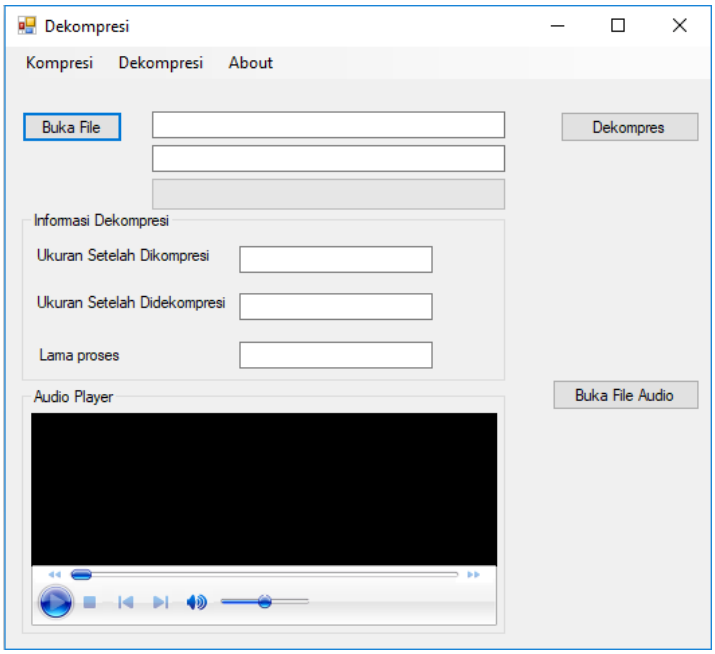
GAMBAR : 4.1. Tampilan Utama

2. Tampilan Menu Kompresi



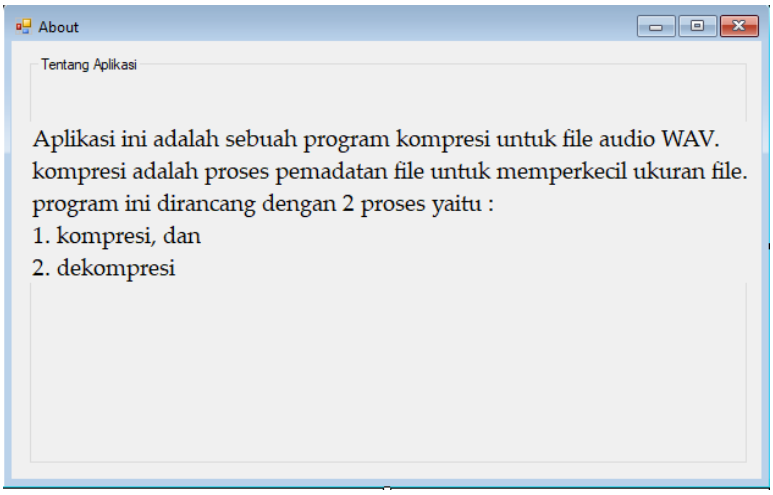
GAMBAR : 4.2. Tampilan Menu Kompresi

3. Tampilan Menu Dekompresi



Gambar : 4.3. Tampilan Menu Dekompresi

4. Tampilan Menu *About*



GAMBAR : 4.4. Tampilan Menu *About*

4.4. *Testing*

Pada tahap ini, sistem akan di uji apakah sudah sesuai dengan fungsi-fungsi yang telah dirancang pada saat tahap analisis dan perancangan. Pengujian ini dilakukan pada *file audio* berformat WAV.

4.4.1. *Testing Black Box*

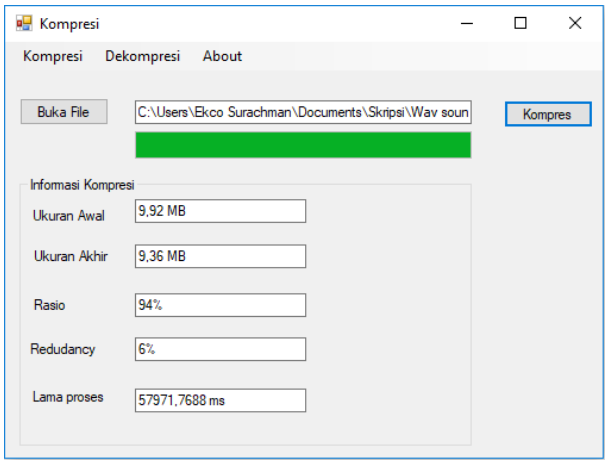
Testing ini dilakukan dengan tujuan untuk menjamin apakah sistem yang dibuat sesuai dengan apa yang diharapkan. Pengujian dibagi tiga yaitu pengujian pada halaman beranda, pengujian pada *form* kompresi dan pengujian pada *form* dekompresi.

4.4.2. Testing Proses kompresi

Untuk melakukan proses kompresi tahap awal yang dilakukan adalah memilih tab Kompresi. Setelah tampilan *form* kompresi muncul maka lakukan langkah-langkah berikut untuk melakukan proses kompresi.

1. Menekan tombol pada Buka *File* untuk membuka *File Dialog*, dan pilih *file audio* (*.*Wav*) sebagai *file* masukan.
2. Menekan tombol kompresi untuk melakukan proses kompresi.

3. Setelah proses selesai aplikasi akan menampilkan informasi hasil kompresi seperti pada gambar 4.5 berikut ini :

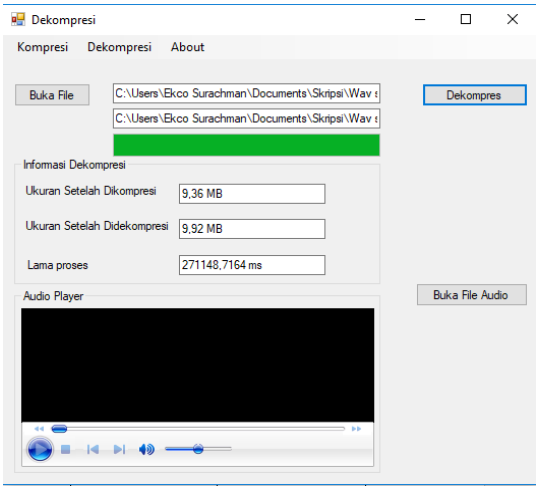


GAMBAR : 4.5. *Testing* proses Kompresi

4.4.3. Testing Proses Dekompresi

Untuk melakukan proses dekompresi tahap awal yang dilakukan adalah memilih tab Dekompresi. Setelah tampilan *form* dekompresi muncul maka lakukan langkah-langkah berikut untuk melakukan proses dekompresi.

1. Buka *file* yang telah dikompresi sebelumnya dan *library*.
2. Selajutnya klik tombol dekompresi untuk melakukan proses dekompresi.
3. Setelah proses selesai aplikasi akan menampilkan informasi hasil dekompresi seperti pada gambar 4.6 berikut ini :



GAMBAR : 4.6. *Testing* proses Dekompresi

4.4.4. Hasil pengujian proses kompresi

Hasil kompresi akan ditentukan dari besar kecilnya persentase rasio yang di dapat dimana semakin kecil rasio maka semakin baik hasil kompresi. Persentase rasio kompresinya dapat dinyatakan sebagai berikut :

1. Rasio 100% - 88% = Kurang baik.
2. Rasio 87% - 50% = Baik.
3. Rasio 49% - 0% = Sangat baik.

TABEL : 4.1. Hasil pengujian proses Kompresi

Nama *File*	Besar Data Sebelum di kompresi (MB)	Besar Data Sesudah di kompresi (MB)	Rasio (%)	*Redudancy* (%)	Waktu kompresi (ms)	Keterangan
Tes 01.wav	9,92	9,36	94	6	57971,7688	Kurang baik
Tes 02.wav	2,13	1,73	81	19	9217,2025	Baik
Tes 03.wav	3,8	2,82	74	26	15058,1261	Baik
Tes 04.wav	2,26	1,84	81	19	10290,2011	Baik
Tes 05.wav	2,47	2,15	87	13	11786,8897	Baik
Tes 06.wav	6,42	5,57	87	13	29436,6094	Baik
Tes 07.wav	14,64	12,88	88	12	83714,2745	Kurang Baik
Tes 08.wav	9,38	6,12	65	35	32863,6474	Baik
Tes 09.wav	2,77	2,48	90	10	14811,3344	Kurang baik
Tes 10.wav	2,2	1,81	82	18	11254,5319	Baik
Tes 11.wav	3,35	2,68	80	20	16156,0255	Baik
Tes 12.wav	1,09	0,97	89	11	6593,1894	Kurang Baik
Tes 13.wav	1,67	1,33	79	21	7027,7307	Baik
Tes 14.wav	1,46	1,15	79	21	6901,376	Baik
Tes 15.wav	6,42	5,57	87	13	27502,8788	Baik

Tes 16.wav	3,84	3,4	89	11	16889,2098	Kurang Baik
Tes 17.wav	5,55	4,45	80	20	23366,8159	Baik
Tes 18.wav	2,77	2,29	83	17	11681,2084	Baik
Tes 19.wav	2,49	1,81	72	28	10266,8812	Baik
Tes 20.wav	2,70	2,42	90	10	13356,2078	Kurang Baik

4.4.5. Hasil pengujian proses Dekompresi

TABEL : 4.2. Hasil pengujian proses Dekompresi

Nama *File*	Besar data Awal (Mb)	Ukuran *File* Setelah Dikompresi (Mb)	Ukuran *File* Setelah Didekompresi (Mb)	Waktu Dekompresi (ms)
Tes 01.wav	9,92	9,36	9,92	271148,7164
Tes 02.wav	2,13	1,73	2,13	38005,5058
Tes 03.wav	3,8	2,82	3,8	76829,4564
Tes 04.wav	2,26	1,84	2,26	47273,8731
Tes 05.wav	2,47	2,15	2,47	61833,4617
Tes 06.wav	6,42	5,57	6,42	157444,7005
Tes 07.wav	14,64	12,88	14,64	374269,0662
Tes 08.wav	9,38	6,12	9,38	134538,9318
Tes 09.wav	2,77	2,48	2,77	53796,4692
Tes 10.wav	2,2	1,81	2,2	45687,9653
Tes 11.wav	3,35	2,68	3,35	58125,8814
Tes 12.wav	1,09	0,97	1,09	18819,1791
Tes 13.wav	1,67	1,33	1,67	24310,8895
Tes 14.wav	1,46	1,15	1,46	22785,4109
Tes 15.wav	6,42	5,57	6,42	124725,3222
Tes 16.wav	3,84	3,4	3,84	69585,4548
Tes 17.wav	5,55	4,45	5,55	113704,1681
Tes 8.wav	2,77	2,29	2,77	43164,3557
Tes 19.wav	2,49	1,81	2,49	43679,5879
Tes 20.wav	2,70	2,42	2,70	52890,2144

4.5. Hasil Analisis

Dari hasil pengujian proses kompresi didapat bahwa rasionya mempunyai besaran antara 65% untuk nilai terkecil dan 94% untuk nilai tertinggi. Jika dicari hasil rasio kompresi

tersebut secara rata-rata adalah sebesar 82,85%. Ini berarti ukuran *file* hasil adalah 0,8285 kali ukuran *file* semula dan pengurangan ukuran *file* sebesar (100% - 82.85%) = 17,15 %.

Dari hasil tersebut juga menunjukkan bahwa persentase kompresi atau dekompresi *file* tidak bergantung pada ukuran *file* melainkan bergantung pada isi data pada *file* tersebut. Semakin banyak perulangan data yang terdapat pada bagian *chunk* data *file* maka rasio kompresi akan semakin rendah.

Dari beberapa pengujian yang dilakukan tingkat kecepatan baik untuk proses kompresi dan dekompresi berbanding lurus dengan ukuran *file* , artinya semakin besar ukuran *file* yang diproses maka semakin lama proses berlangsung. Dari hasil 20 ujicoba yang dilakukan didapat hasil bahwasanya metode *Huffman* memiliki rata-rata rasio sebesar 82,85 % ini membuktikan bahwasanya metode *Huffman* cukup baik dalam mengkompresi *file audio* WAV.

5. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil pembahasan beserta penelitian yang telah dilakukan, maka dapat diambil beberapa kesimpulan, diantaranya sebagai berikut :

1. Aplikasi yang dirancang untuk kompresi *file audio* WAV dengan menggunakan metode *Huffman* telah berhasil direalisasikan dan bekerja dengan baik.
2. Berdasarkan hasil pengujian proses kompresi didapat bahwa rasio mempunyai besaran antara 65 % untuk nilai terkecil dan 94% untuk nilai tertinggi. Dengan rata-rata rasio sebesar 82.85%. Ini berarti ukuran *file* hasil adalah 0,8285 kali ukuran *file* semula dan pengurangan ukuran *file* sebesar 17.15 %. Sedangkan untuk tingkat kecepatan proses kompresi dan dekompresi berbanding lurus dengan ukuran *file* , artinya semakin besar ukuran *file* yang diproses maka semakin lama proses berlangsung.

5.2. Saran

Dengan adanya kesimpulan diatas, ada beberapa saran yang dapat dikemukakan sebagai bahan pertimbangan lebih lanjut guna meningkatkan produktifitas kerja dari aplikasi kompresi ini.

1. Melakukan kompresi dengan *file* lain misalnya seperti *video*, gambar, dan *text*.
2. Penelitian ini dapat di kembangkan lagi dengan memperbandingan metode *Huffman* dengan metode lainnya seperti *Shannon-fano, RLE, Half-Byte, LZW*.
3. Diharapakan untuk penelitian kedepannya hasil kompresi dapat langsung dimainkan tanpa melakukan proses dekompresi lagi.
4. Penelitian ini akan lebih baik lagi apabila menggunakan algoritma *Rice coding* sebagai metode algoritmanya.

6. DAFTAR PUSTAKA

Aburas, A.A., Rehiel,S.A (2008). *Fingerprint patterns recognition system using Huffman coding. Proceedings of the World Congress on Engineering*, London: WCE,Vol III.

Andysah P.,& Utama.S. 2016. *Implementasi teknik kompresi teks Huffman*. Fakultas Ilmu Komputer Universitas Pembanguan Panca Budi. Jurnal Informatika Vol.10, No.2.

Binanto. 2010. *Multimedia Digital Dasar Teori +Pengembangan*. Penerbit Andi. Yogyakarta.

David Huffman. 1952 . *The Huffman Algorithm*.

Dedi, D., Rizky P., & Nurul.H. 2018. *Kombinasi gifshuffle, enkripsi AES dan kompresi data huffman untuk meningkatkan keamanan data*. Universitas Lampung. Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK).Vol.05, No.4, Hlm, 389-394.

Deswintiani Sihotang.2017. *Perancangan aplikasi keamanan data text dengan metode idea dan kompresi menggunakan algoritma Huffman*.Teknik Informatika STMIK Budidarma Medan. Majalah Ilmiah INTI Vol. XII, No.1.

Devie R. Suchendra, Sandra W. 2012. *Implementasi kompresi data text menggunakan Huffman coding* .Program Studi Teknik Informatika STMIK LPKIA. JURNAL LPKIA, Vol.1 No.1.

Howe, D. (1993). Free on-line dictionary of computing. Diakses dari http://www.foldoc.org.

Karmela, S.M.W.,Willy, S.R.,& Antonius,R.C. 2010. *Aplikasi Player untuk Menjalankan File Wave yang terkompresi dengan Metode Huffman*. Fakultas Teknik Universitas Kristen Duta Wacana. Jurnal Informatika, Vol.6, No.1.

Kharisma, M., & Karpen.2017. *Rancang bangun aplikasi kompresi dan dekompresi pada citra digital menggunakan metode Huffman*. STMIK Amik Riau. PROCESSOR Vol.12, No.1.

Kurniawan, Erick. 2010. *Cepat Mahir Visual Basic 2010.* Yogyakarta: Andi.

Ramadhani C. 2015. *Dasar Algoritma & Struktur Data dengan Bahasa JAVA, 1e*. Yogyakarta : Andi Offset.

Rolly Yesputra, 2017. *Belajar visual basic. Net Dengan visual studio 2010*. Penerbit Royal Asahan Press. Sumatra Utara.

Sayood, K. 2005. *Introduction to data compression. Morgan Kaufmann,* San Fransisco.

Salomon, D. (2004). *Data compression complete reference 3rd Edition*. Springer, NewYork.

Salomon. 2007. *The Compression Algorithm* ,Data Compression reference Center.

Victor Amrizal.2010. *Implementasi Algoritma Kompresi Data Huffman Untuk Memperkecil Ukuran File MP3 Player*. Universitas Islam Negeri Syarif Hidayatullah Jakarta. Jurnal Sistem Informasi, 3(2), 2010, 1 – 14.