BAB II

LANDASAN TEORI

2.1 *E-budgeting*

Pengertian *E-budgeting* merupakan sistem informasi keuangan berbasis web yang digunakan untuk memfasilitasi proses penyusunan anggaran. Melalui *E-budgeting*, perusahaan dapat melakukan pengendalian anggaran dengan lebih baik karena proses penyusunan anggaran dapat dilakukan secara lebih transparan, efektif dan akuntabel.

Keterbukaan atau transparansi menjadi tujuan utama penerapan sistem informasi penyusunan anggaran ini. Setiap pihak yang berperan penting dalam perusahaan atau *stakeholder* bisa mengakses data-data anggaran karena dokumentasi penyusunan anggaran telah tercatat dan tersimpan otomatis dalam sistem. Aplikasi program sudah disusun sedetail mungkin, sehingga setiap divisi di perusahaan bisa memasukan rincian anggaran dengan cermat.

Terdapat 5 tujuan mengapa suatu perusahaan perlu menerapkan sistem informasi ini yaitu:

a) Memudahkan penyusunan anggaran

Dengan sistem yang telah terprogram baik, proses penyusunan anggaran lebih mudah dan cepat kerena semua detail informasi sudah terformat dalam aplikasi.

b) Meningkatkan kualitas anggaran

Sistem yang terkomputerisasi seperti ini mendorong akurasi anggaran yang disusun sesuai alokasi belanja masing-masing divisi. Karena semua data dimasukan

secara cermat, kualitas anggaran meningkat dan dapat dipertanggungjawabkan di kemudian hari.

c) Meningkatkan Transparansi Anggaran

Penerapan sistem ini bertujuan untuk meningkatkan transparansi anggaran perusahaan, sehingga semua *stakeholder* dapat mengakses, memantau dan mengawasi pembuatan serta pemakaian anggaran tersebut.

d) Terintegrasi dengan sistem informasi lainnya

Ketika seluruh sistem informasi manajemen keuangan perusahaan sudah terintegrasi, proses perencanaan kegiatan perusahaan pun akan lebih mudah dan lebih cepat dilakukan.

e) Memudahkan penyusunan laporan

Semua data keuangan perusahaan sudah tersimpan baik dalam sistem informasi yang ada. Maka penyusunan berbagai jenis laporan yang dibutuhkan dapat dieksekusi dengan mudah, cepat dan cermat.

2.1.1 Prinsip Dasar Rancangan Anggaran

Dalam penyusunan perencanaan anggaran ini, ada beberapa prinsip dasar yang perlu dipenuhi. Tujuannya agar anggaran yang dirancang dapat disusun dan dilaksanakan sesuai dengan rencana. Prinsip dasar rancangan anggaran yang dimaksud, antara lain adalah sebagai berikut:

a) *Management involvement*

Keterlibatan manajemen perusahaan dalam menyusun rencana anggaran. Dengan begitu, artinya manajemen punya komitmen dalam mencapai target jangka pendek dan jangka panjang perusahaan.

b) *Organizational adaption*

Rencana keuangan harus disusun sesuai dengan struktur organisasi perusahaan, dimana ada ketegasan wewenang dan tanggungjawab.

c) *Responsibility accounting*

Perlu ada sistem *Responsibility Accountung* supaya rencana keuangan dapat dilaksanakan dengan baik. Polanya dapat disesuaikan dengan pertanggungjawaban manajemen keuangan perusahaan.

d) *Goal orientation*

Dalam menyusun anggaran perlu adanya tujuan yang realistis. Dengan adanya tujuan yang realistis maka perusahaan dapat mengembangkan bisnisnya secara jangka panjang.

e) *Full Communication*

Perencanaan dan pengendalian keuangan dapat berjalan dengan efektif apabila antar tingkat manajemen memiliki pemahaman yang sama tentang tanggungjawab dan target yang akan dituju.

f) *Realistic Expectation*

Dalam perencanaa keuangan perlu adanya ekspektasi yang realistis untuk menjaga kelangsungan pertumbuhan perusahaan. Manajemen perlu menghindari konservatisme dan optimisme yang berlebihan.

g) *Timeliness*

Laporan mengenai realisasi keuangan harus diterima manajer yang berkompeten agar informasi tersebut dapat digunakan sebagai dasar pengambilan kebijakan oleh manajemen.

h) *Flexible apllication*

Fleksibel maksudnya adalah perencanaan keuangan tidak boleh kaku. Dalam perencanaan keuangan perlu ada celah untuk perubahan sesuai dengan situasi dan kondisi.

i) *Reward & punishment*

Manajemen perlu melakukan penilaian kinerja manajer dari perencanaan keuangan yang dilakukan. Sehingga ada *reward* atau justru *punishment* yang diberikan berdasarkan kinerja mereka secara transparan.

2.2 Teknologi Perangkat Lunak

2.2.1 PHP

Menurut Kevin Tantroe, Peter Macintyre dan Rasmus Lerdorf dalam Programming PHP (2013, 1), PHP adalah bahasa yang sederhana namun kuat yang dirancang untuk membuat konten HTML.

2.2.1.1 Struktur Leksikal

Menurut Kevin Tantroe, Peter Macintyre dan Rasmus Lerdorf dalam Programming PHP (2013, 1), Struktur leksikal bahasa pemrograman adalah seperangkat aturan dasar yang mengatur cara penulisan bahasa pemograman. Ini adalah sintaks level terendah dari bahasa dan menentukan hal-hal seperti nama variabel terlihat, karakter yang digunakan untuk sebuah komentar, dan penulisan perintah pemograman yang dipisahkan dengan perintah pemograman yang lainnya.

2.2.1.2 Tipe Data

PHP menyediakan delapan jenis nilai, atau tipe data. Empat adalah tipe skalar (satuan nilai): *integer, floating-point, string,* dan *boolean*. Dan dua adalah

tipe gabungan (koleksi): *array* dan *objects*. Dua yang tersisa adalah tipe khusus: Resources dan NULL.

2.2.1.3 Variabel

Variabel dalam PHP adalah pengenal yang diawali dengan tanda dolar ($). Sebagai contoh:

$name $Age $_debugging

Suatu variabel dapat memiliki nilai dari tipe apa pun. Pada saat mengganti nilai variabel dengan nilai lain yang berbeda mengetik:

```
$what = "Fred";
$what = array("Fred", 35, "Wilma");
```

Tidak ada sintaks eksplisit untuk mendeklarasikan variabel dalam PHP, dimana pertama kali nilai variabel diatur atau variabel dibuat. Dengan kata lain, pengaturan nilai ke variabel juga berfungsi sebagai deklarasi. Sebagai contoh, ini adalah program PHP lengkap yang valid:

```
$day = 60 * 60 * 24;
echo "There are {$day} seconds in a day.\n";
```

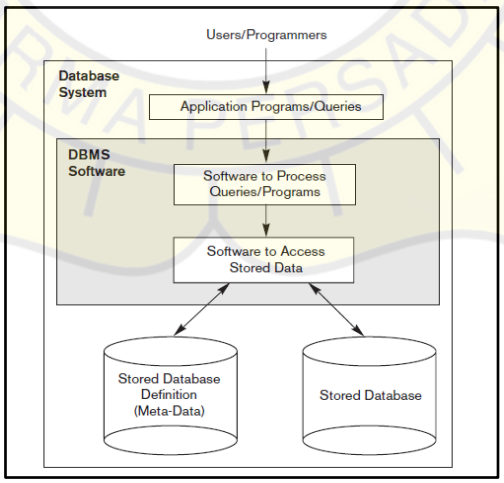
2.2.2 Basis Data

Basis data dan teknologi basis data memiliki dampak yang besar terhadap meningkatnya pengguna komputer. Adil untuk mengatakan bahwa basis data memainkan peranan penting di hampir semua area, di mana computer tersebut digunakan termasuk dunia usaha, perdagangan elektronik, teknik, kedokteran, genetika, hukum, pendidikan, dan ilmu perpustakaan. Kata basis data sangat umum digunakan, maka harus dimulai dengan mendefinisikan apa arti dari basis data itu sendiri. Menurut Ramez Elmasri, Shamkant B. Navanthe dalam Fundamentals of

Basis data Systems 6th edition (2011, 4) Basis data adalah kumpulan data yang terkait. Dengan data, fakta yang diketahui, direkam dan memiliki makna yang implisit.

2.2.2.1 *Database Management System*

Menurut Ramez Elmasri, Shamkant B. Navanthe dalam Fundamentals of Basis data Systems 6th edition (2011, 5) *Basis data Management System* (DBMS) adalah kumpulan program yang memungkinkan pengguna untuk membuat dan memelihara basis data. DBMS adalah sistem perangkat lunak tujuan umum yang memfasilitasi proses mendefinisikan, membangun, memanipulasi, dan berbagi basis data di antara berbagai pengguna dan aplikasi. Mendefinisikan sebuah basis data melibatkan menentukan tipe data, struktur, dan batasan data yang akan disimpan dalam basis data. Definisi basis data atau informasi deskriptif juga disimpan oleh DBMS dalam bentuk katalog atau kamus basis data; itu disebut *meta-data*. Membangun basis data adalah proses penyimpanan data pada beberapa medium penyimpanan yang dikendalikan oleh DBMS.



Gambar 2.1 Lingkup Sistem Basis Data Yang Disederhanakan (*Elmasri & B.Navathe*, 2010)

2.2.2.2 *SQL*

Bahasa SQL dapat dianggap sebagai salah satu alasan utama keberhasilan komersial dari basis data relasional. Karena itu menjadi standar untuk database relasional, pengguna kurang peduli tentang migrasi aplikasi database mereka dari jenis sistem basis data lainnya misalnya jaringan atau sistem hirarki ke basis data relasional. Hal ini karena jika pengguna menjadi tidak puas dengan produk DBMS relasional tertentu yang digunakan, makan pengguna mengkonversi ke produk DBMS relasional yang lain tidak diharapkan terlalu mahal dan memakan waktu karena kedua sistem mengikuti standar bahasa yang sama.

SQL adalah kepanjangan dari *Structured Query Language,* yang awalnya disebut SEQUEL *(Structured English Query Language)* yang didesain dan diimplementasi pada IBM Research sebagai tampilan antar muka eksperimental sistem basis data relasional *yaitu SYSTEM R.SQL* yang lebih dikenal sekarang sebagai komersial DBMS relasional. SQL adalah bahasa basis data yang komprehensif dengan memiliki pernyataan untuk mendefinisi, *queries*, dan pembaruan data.

2.2.2.3 *PHP Data-Objects (PDO)*

Pada pemograman PHP ada dua cara untuk mengakses basis data yaitu menggunakan *basis data-spesific extension* dan *basis data-independent PDO.* Ada kelebihan dan kekurangan menggunakan untuk setiap pendekatan akses basis data. Dengan mengguna *basis data-spesific extension* kode pemograman harus terkait erat dengan basis data yang digunakan, contohnya nama fungsi, parameter, dan penanganan error pada basis data MySQL berbeda dengan basis data lainnya. Jika ingin merubah sistem basis data dari basis data MySQL ke basis data Postgre akan

merubah kode secara signifikan. PDO menyembunyikan *basis data-spesific* menggunakan *abstraction layer*, sehingga perpindahan antara sistem basis data yang satu dengan lain sama mudahnya seperti merubah satu baris kode pemograman atau pada file *php.ini*.

2.2.3 Android

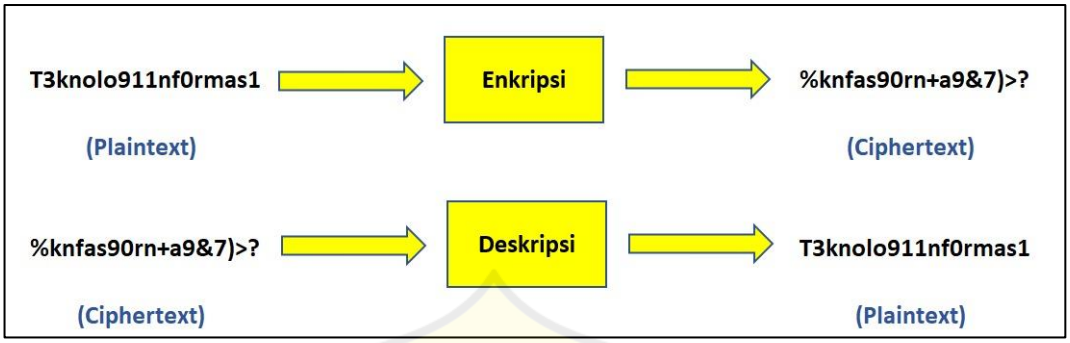
Android adalah sebuah sistem operasi untuk *Smartphone* dan Tablet. Android dikembangkan oleh perusahaan – perusahaan yang tergabung dalam sebuah konsorsium bernama Open Handset Alliance (OHA). OHA dipimpin oleh Google dan didirikan bersama 34 perusahaan lainnya, dengan tujuan untuk mengembangkan teknologi mobile device. Sekarang ini, anggota konsorsium sudah berjumlah 84 perusahaan yang bergerak di berbagai bidang seperti pembuat device, semikonduktor, pembuat aplikasi, komersialisasi dan mobile operator.

Android merupakan sistem operasi yang bersifat *open source* (sumber terbuka). Disebut *open source* karena *source code* (kode sumber) dari sistem operasi android dapat dilihat, di download dan dimodifikasi secara bebas. Paradigma *open source* ini memudahkan pengembang teknologi Android, karenaa semua pihak yang tertarik dapat memberikan kontribusi, baik pada pengembangan sistem operasi maupun aplikasi.

2.2.4 Kriptografi

Kriptografi merupakan merupakan studi metode untuk mengirim pesan secara rahasia (yaitu, dienkripsi atau disamarkan) sehingga hanya penerima yang dimaksud dapat menghapus penyamaran dan membaca pesan (atau menguraikannya). Kata krriptografi memiliki *etimologi*, yaitu kripto dari Bahasa Yunani, artinya tersembunyi, dan graphein, artinya menulis. Pesan asli disebut

plaintext, dan pesan tersamar disebut ciphertext. Pesan yang telah dienkapsulasi dan dikirim, disebut *cryptogram*.



Gambar 2.2 Contoh Penerapan Kriptografi

Proses mengubah plaintext menjadi ciphertext disebut enkripsi atau penyandian. Sedangkan proses mengubah kembali ciphertext menjadi plaintext, yang dilakukan oleh penerima yang memiliki pengetahuan untuk menghapus penyamaran, disebut dekripsi atau menguraikan. Seorang yang terlibat dalam proses kriptografi disebut seorang *cryptographer*. Di sisi lain, studi tentang teknik matematika yang berusaha merusak metode kriptografi disebut *cryptanalysis*. Mereka yang berlatih *cryptanalysis* disebut *cryptanalysts*.

2.2.5 *Advanced Encryption Standard* (AES)

Advanced Encryption Standard (AES) adalah algoritma kriptografi yang bisa digunakan untuk mengamankan data. Algoritma AES ini bekerja pada blok *ciphertext* simetrik yang dapat mengenkripsi (*encipher*) dan dekripsi (*dechipher*) informasi. Algoritma AES ini menggunakan algoritma Rijndael dengan kunci kriptografi 128, 192, dan 256 bits untuk mengenkripsi dan mendekripsi suatu data. Maka dari itu algoritma ini dikenal juga sebagai AES-128, AES-192, dan AES-256.

AES sendiri menggunakan struktur SPN (*Subsitution Permutation Network*) yang memiliki derajat palalelisme yang lebih besar. Kelemahan pada stuktur SPN ini pada umumnya adalah berbedanya struktur enkripsi dan dekripsi sehingga diperlukan dua algoritma yang berbeda untuk enkripsi dan dekripsi. AES memiliki blok masukan dan keluaran serta kunci 128 bit. Untuk tingkat keamanan yang lebih tinggi, AES dapat menggunakan kunci 192 dan 256 bit. Setiap masukan 128 bit *plaintext* dimasukan ke dalam *state* yang berbentuk bujur sangkar berukuran 4x4 *byte*. *State* ini di-XOR dengan *key* dan selanjutnya diolah 10 kali dengan subtitusi-transformasi linear-*Addkey* untuk memperoleh *ciphertext*.

Berikuti ini adalah operasi Rijndael (AES) yang menggunakan 128 bit kunci:

a) Ekpansi kunci utama (mulai dari 128 bit menjadi 1406 bit)
b) Pencampuran *subkey*
c) Ulang dari i=1 sampai i=10 Transformasi: *ByteSub* (subtitusi per *byte*) *ShiftRow* (pergeseran *byte* perbaris) *MixColumns* (operasi perkalian GF(2^8) per kolom) (Publications, 2001)
d) Pencampuran *subkey* (dengan XOR)
e) Transformasi: *ByteSub* dan *ShiftRow*
f) Pencampuran *subkey*

2.2.5.1 Representasi Data

Hasil dari input dan output algoritma AES ini terdiri dari urutan data sebesar 128 bit yang sudah terbentuk dalam satu kelompok yang selanjutnya disebut sebagai blok data atau *plaintext* yang kemudian akan dilakukan proses enkripsi

menjadi *chiphertext*. *Chiper key* dari AES bergantung pada panjang bit yang digunakan.

Bit-bit tersebut diberi nomor urut mulai dari 0 sampai dengan n-1 dimana n adalah nomor urutan. Urutan data 8 bit secara beruntun disebut sebagai *byte* sebagai unit dasar dari operasi yang akan dilakukan pada blok data.

Algoritma AES untuk data sepanjang 128 bit akan dibagi menjadi *array byte* yang terdiri dari 8 bit data input yang saling berurutan. Berikut representasi *array byte* dalam bentuk:

$a_0 a_1 a_2 \dots a_{15}$

Dimana:

$a_0 = \{input_0, input_1, \dots, input_7\}$

$a_1 = \{input_8, input_9, \dots, input_{15}\}$

$a_{15} = \{input_{120}, input_{121}, \dots, input_{127}\}$

$a_n = \{input_{8n}, input_{8n+1}, \dots, input_{8n+7}\}$

Operasi algoritma AES dilakukan pada sebuah *state* yang memiliki *array byte* dua dimensi. Setiap *state* memiliki jumlah baris yang tetap yaitu 4, sedangkan jumlah kolom tergantung pada besarnya blok data. Baris pada *state* mempunyai indeks nomor *row* (r) dimana $0 \leq r < 4$. Sedangkan kolom mempunyai indeks *column* (c) dimana $0 \leq c < Nb$. Nb adalah besarnya blok data dibagi dengan 32.

Permulaan pada saat input bit akan disusun menjadi suatu *array byte* dimana panjang dari *array byte* yaitu 8 bit data. *Array byte* inilah yang kemudian akan dimasukan atau di *copy* kedalam *state* dengan urutan (Fachrurrozi, Muhammad Farid, 2006):

$s[r,c] = in[r + 4c]$ untuk $0 \leq r < 4$ dan $0 \leq c < Nb$

sedangkan dari *state* yang akan di *copy* ke output dengan urutan:

out[r + 4*c*] = *s*[*r*,*c*] untuk $0 \leq r < 4$ dan $0 \leq c < Nb$

2.2.5.2 Operasi Aljabar

Metode AES juga merupakan proses operasi matematika karena setiap tahapan atau langkah transformasinya melibatkan *state* dengan unit dasar operasi AES yaitu *byte* dan setiap *byte* merupakan elemen *finite field GF* (2^8) dengan sebuah himpunan modulo 8 yang didefinisikan oleh operasi penjumlahan dan perkalian.

Elemen *finite field* adalah elemen dari *field* yang memiliki sifat *ring* komutatif. Dalam hal ini untuk semua *finite field* yang memiliki p^n dimana *p* merupakan bilangan prima dan *n* merupakan bilangan bulat $n \geq 1$ sama dinotasikan dengan *GF* (p^n). Elemen *GF* (2^8) juga merupakan *ring* komutatif yang memiliki sifat-sifat berikut (Wagstaff, 2003):

a) Grup (G, +), Grup sendiri memenuhi sifat tertutup pada operasi penjumlahan, asosiatif pada operasi penjumlahan, memiliki elemen identitas, mempunyai invers, dan grup abelian.
b) *Ring* (G, +, *), *Ring* juga memenuhi sifat tertutup pada operasi perkalian, asosiatif pada operasi perkalian, dan distributif.

2.2.5.3 Field GF (2^8)

Terdapat beberapa cara yang berbeda untuk bisa direpresentasikan elemen dari *finite field* yaitu dengan cara polinomial, bit, maupun heksadesimal. Untuk semua pangkat *n* adalah bilangan prima yang memiliki satu *finite field* (Farchrurrozi, Muhammad Farid, 2006). Sehingga *GF* (2^8) dan ASCII merupakan isomorfisme atau homomorfisme yang memiliki fungsi satu-satu. Sedangkan

homomorfirsme adalah fungsi dari *ring* ke *ring* lain yang mempunyai sifat *f* (*a* + *b*) = *f* (*a*) + *f* (*b*) dan *f* (*ab*) = *f* (*a*) *f* (*b*) (Wagstaff, 2003). Jadi ketika elemen tersebut direpresentasikan maka akan memiliki pengaruh yang kuat dalam implementasi yang kompleks. Dalam hal ini akan direpresentasikan *ring* atas polinomial. Jika b merupakan suatu nilai dari 0 atau 1 maka terbentuk ukuran *byte* dari urutan bit $b_7 + b_6 + b_5 + b_4 + b_3 + b_2 + b_1 + b_0$ (koefisien *binary*) sehingga dapat dituliskan pada persemaan berikut:

$$b\,(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$$

Dari persamaan diatas dapat diketahui bahwa pangkat tertinggi dari polionomial GF (2^8) adalah x^7.

a) Penjumlahan

Penjumlahan dari dua elemen *finite field* dapat didefinisikan dengan operasi XOR (penjumlahan 2 elemen dengan modulo 2) per bit. Sehingga penyederhanaannya merupakan operasi identik. Berikut sebagai contoh penerapan: 30 ⊕ *d*4 =

00110000

<u>11010100</u> ⊕

<u>11100100</u>

$(x^5 + x^4) + (x^7 + x^6 + x^4 + x^2) = (x^7 + x^6 + x^4 + x^2)$

Bentuk biner dari $(x^7 + x^6 + x^4 + x^2)$ adalah 11100100 dan dalam heksadesimal diperoleh *e*4. Semua kondisi yang penting dalam menyelesaikan operasi di atas merupakan bagian dari grup abelian.

b) Perkalian

Perkalian element GF (2^8) adalah perkalian yang direpresentasikan dalam polinomial dengan modula polinomial m(x) yang *irreducible* dari polinomial pangkat 8 (Vincent, 2003). *Irreducible* adalah polinomial yang hanya mempunyai faktor 1 dan bilangan itu sendiri, sehingga dapat dituliskan persamaan sebagai berikut:

$m(x) = (x^8 + x^4 + x^3 + x + 1)$

Atau 11B dalam bentuk heksadesimal dan bentuk desimal adalah 283, sebagai contoh:

$57 \cdot 84$

$= [01010111] \cdot [10000100]$

$= (x^6 + x^4 + x^2 + x + 1)\ (x^7 + x^2)\ mod$

$(x^8 + x^4 + x^3 + x + 1)$

$= (x^{13} + x^{11} + x^9 + x^8 + x^7 + x^8 + x^6 + x^4 + x^3 + x^2) mod$

$(x^8 + x^4 + x^3 + x + 1)$

$= (x^{13} + x^{11} + x^9 + x^7 + x^6 + x^4 + x^3 + x^2) mod$

$(x^8 + x^4 + x^3 + x + 1)$

$= (x^{13} + x^{11} + x^9 + x^8 + x^7 + x^6 + x^2 + x + 1) mod$

$(x^8 + x^4 + x^3 + x + 1)$

$= x^{13} + x^{11} + x^9 + x^7 + x^6 + x^4 + x^3 + x^2 + 1$

Dari bentuk polinomial mod 8 tersebut dirubah ke biner menjadi 11011101 yang dalam bentuk heksadesimalnya adalah *dd*.

c) Perkalian dengan variabel *x*

Jika dituliskan perkalian *b*(*x*) sebagai berikut:

$x \cdot b(x) = b_8 x^8 + b_7 x^7 + b_6 x^6 + b_5 x^5 + b_4 x^4 + b_3 x^3 + b_2 x^2 + b_1 x + b_0$

perkalian $x \cdot b(x)$ dapat diwujudkan sebagai *leftshift* atau pergeseran bit ke kiri yang diikuti oleh XOR kondisional dengan (1b), jika b_8= 1 maka XOR dilakukan, namun jika b_8= 0 maka tidak dilakukan XOR. *Exclusive* OR kondisional tersebut tidak lain adalah operasi modulo dengan *m*(*x*). Serangkaian *left shift* yang disusul operasi XOR tersebut dapat digunakan untuk perkalian antara elemen *finite field*. Operasi $x \cdot b(x)$ dinotasikan sebagai *xtime* (Vincent, 2003). Sebagai contoh adalah sebagai berikut:

$'57' \cdot '57' = 'FE'$

$'57' \cdot '02' =$ *x time* $(57) = 'AE'$

$'57' \cdot '04' =$ *x time* $(AE) = '47'$

$'57' \cdot '08' =$ *x time* $(47) = '8E'$

$'57' \cdot '10' =$ *x time* $(8E) = '07'$

$'57' \cdot '57' = '57' \cdot ('01' \oplus '02' \oplus '10') = '57' \oplus 'AE' \oplus '07' = 'FE'$

2.2.5.4 Koefisien Polinom Pada GF (2^8)

Direpresentasikan polinomial yang didefinisikan dengan koefisien GF (2^8) sebagai persamaan berikut:

$a(x) = a_3x^3 + a_2x^2 + a_1x + a_0$

sehingga didapat bentuk koefisien sebagai berikut $[a_3, a_2, a_1, a_0]$. Polinomial ini berbeda dengan polinomial pada *finite field* yang sebelumnya sebagai polinomial koefisien biner. Pada polinomial ini akan dioperasikan perkalian dengan polinomial yang beberda, akan tetapi bentuk polinomialnya sama yaitu berderajat 4.

$b(x) = b_3x^3 + b_2x^2 + b_1x + b_0$

Definisi dari kedua polinomial diatas dioperasikan sebagai XOR antara persamaan tersebut. Operasi XOR ini berkoresponden antara pangkat pada variabel *x* dapat dilihat pada persamaan berikut:

$a(x) + b(x) = (a_3 \oplus b_3)x^3 + (a_2 \oplus b_2)x^2 + (a_1 \oplus b_1)x + (a_0 \oplus b_0)$

Didefinisikan $a(x) + b(x) = c(x)$ sehingga menghasilkan persamaan sebagai berikut:

$c(x) = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$

didapat dengan cara:

$c_0 = a_0 \cdot b_0$

$c_1 = a_1 \cdot b_0 \oplus a_0 \cdot b_1$

$c_2 = a_2 \cdot b_0 \oplus a_1 \cdot b_1 \oplus a_0 \cdot b_2$

$c_3 = a_3 \cdot b_0 \oplus a_2 \cdot b_1 \oplus a_1 \cdot b_2 \oplus a_0 \cdot b_3$

$c_4 = a_3 \cdot b_1 \oplus a_2 \cdot b_2 \oplus a_1 \cdot b_3$

$c_5 = a_3 \cdot b_2 \oplus a_2 \cdot b_3$

$c_6 = a_3 \cdot b_3$

Hasil dari *c*(*x*) diatas masih dalam bentuk 4 *byte*, maka selanjutnya yaitu *c*(*x*) dimodularkan dengan polinomial berderajat 4. Pada algoritma AES diberikan polionomial $x^4 + 1$ menjadi:

$x^i mod\ (x^4+1) = x^{i\ mod\ 4}$

Operasi modulo dari *a*(*x*) dan *b*(*x*) menghasilkan sebuah *d*(*x*) yang direpresentasikan pada persamaan berikut:

$d(x) = d_3x^3 + d_2x^2 + d_1x + d_0$

Dengan hasil:

$d_0 = a_0 \cdot b_0 \oplus a_3 \cdot b_1 \oplus a_2 \cdot b_2 \oplus a_1 \cdot b_3$

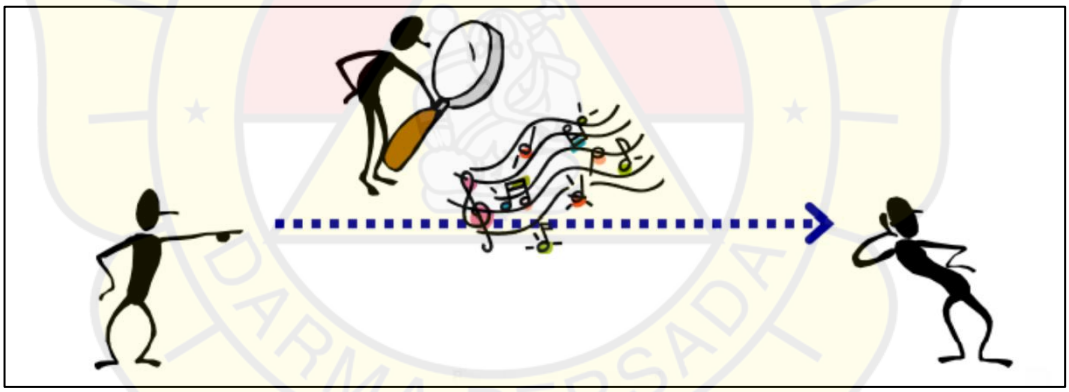
$d_1 = a_1 \cdot b_0 \oplus a_0 \cdot b_1 \oplus a_3 \cdot b_2 \oplus a_2 \cdot b_3$

$d_2 = a_2 \cdot b_0 \oplus a_1 \cdot b_1 \oplus a_0 \cdot b_2 \oplus a_3 \cdot b_3$

$d_3 = a_3 \cdot b_0 \oplus a_2 \cdot b_1 \oplus a_1 \cdot b_2 \oplus a_0 \cdot b_3$

2.2.6 Steganografi

Steganografi merupakan studi metode untuk menyembunyikan pesan dengan suatu cara sehingga selain si pengirim dan si penerima, tidak ada seorangpun yang mengetahui atau menyadari bahwa ada suatu pesan rahasia (tidak terlihat bahwa ada pesan tersembunyi). Kata steganografi berasal dari bahasa Yunani yaitu steganos yang berarti penyamaran atau penyembunyian dan graphein atau graptos yang berarti tulisan sehingga secara keseluruhan artinya adalah tulisan yang disembunyikan.



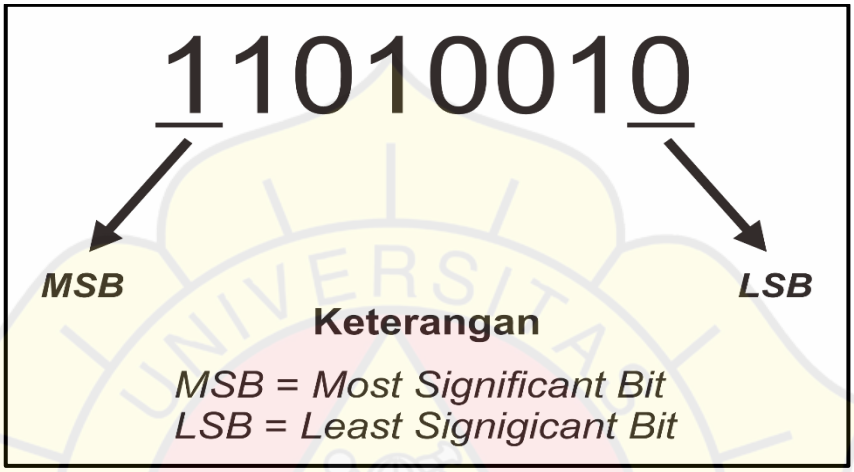
Gambar 2.3 Contoh Penerapan Steganografi

Meskipun memiliki tujuan yang sama dengan kriptografi, keduanya merupakan hal yang berbeda. Pada kriptografi, pesan dikodekan sehingga orang lain tidak mengenali pesan tersebut, sedangkan steganografi menyembunyikan keberadaan pesan sehingga tidak disadari keberadaannya oleh orang lain.

2.2.6.1 *Least Significant Bit* (LSB)

Terdapat beberapa cara menerapkan metode steganografi pada file MP3, salah satunya dengan cara mengganti atau menambahkan bit. Penggantian bit ini

dikenal dengan nama *Least Significant Bit* (LSB). Cara ini biasa digunakan dalam teknik digital steganografi yaitu mengganti LSB input setiap samplingnya dengan data yang dikodekan. Dengan metode ini keuntungan yang didapatkan adalah ukuran pesan yang disisipkan relatif besar, namun berdampak pada hasil audio yang berkualitas kurang dengan banyaknya *noise* (suara yang mengganggu).



Gambar 2.4 Perbedaan MSB dan LSB

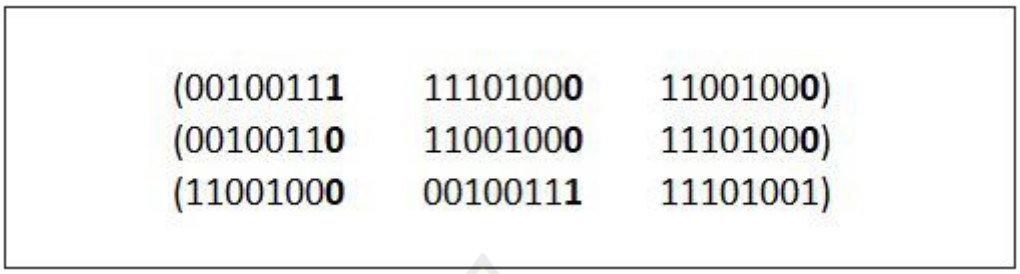
Untuk menjelaskan algoritma LSB ini, digunakan citra digital sebagai *cover-object*. Pada setiap byte terdapat bit yang paling kurang berarti, misalnya pada byte 0001100**1**, maka bit LSB dari byte tersebut adalah **1**. Untuk melakukan penyisipan pesan, bit yang paling cocok untuk diganti dengan bit pesan adalah bit LSB, sebab pengubahan bit tersebut hanya akan mengubah nilai byte-nya menjadi satu lebih tinggi atau satu lebih rendah.

Sebagai contoh, perhatikan bit berikut ini dari *cover-object* yang digunakan pada audio MP3:

(00100111	11101001	11001000)
(00100111	11001000	11101001)
(11001000	00100111	11101001)

Gambar 2.5 Original Bit

Pesan yang akan disisipkan adalah karakter "A", yang nilai biner-nya adalah **10000001**, maka akan menghasilkan *stego-object* dengan urutan bit sebagai berikut:



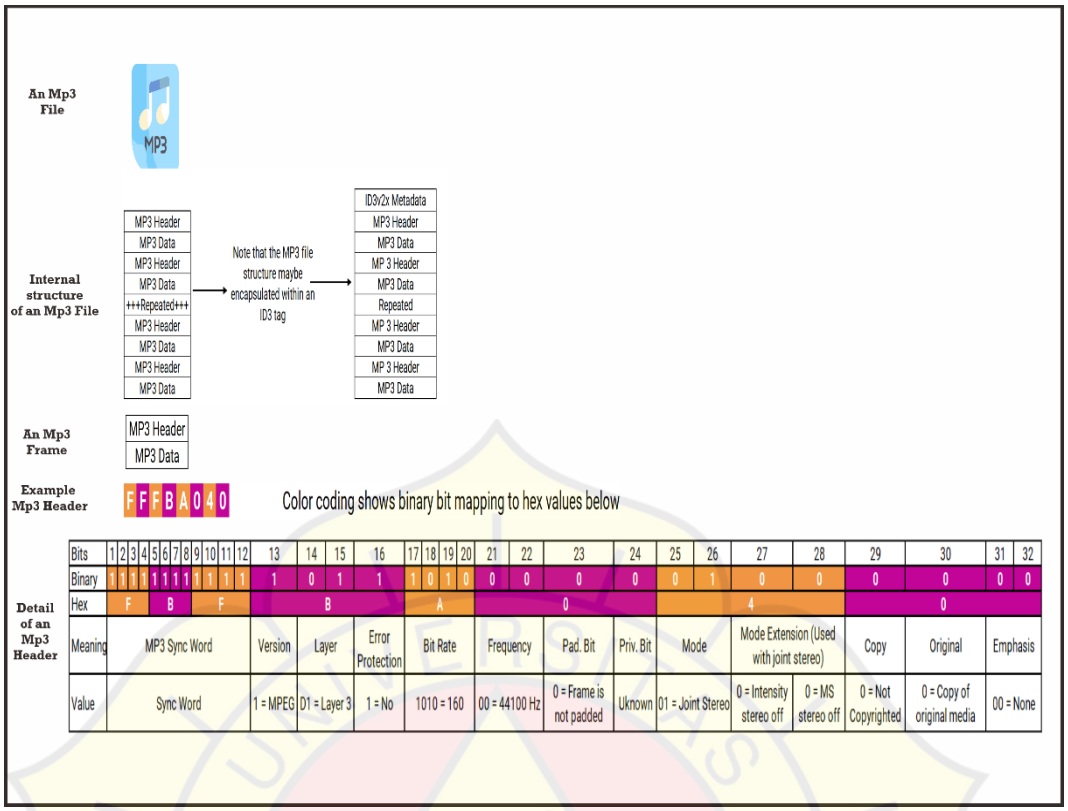
Gambar 2.6 Bit Setelah Enkripsi

Ada dua jenis teknik yang dapat digunakan pada algoritma LSB, yaitu penyisipan pesan secara sekuensial dan secara acak. Sekuensial berarti pesan rahasia disisipkan secara berurutan dari data titik pertama yang ditemukan pada file MP3. Sedangkan acak berarti penyisipan pesan rahasia dilakukan secara acak pada file MP3, dengan memasukan kata kunci (*Stego-key*).

2.2.7 *Motion Picture Expert Group*-3 *Audio Layer* 3 (MPEG 1 Audio *Layer* 3)

MPEG-1 *Audio Layer* 3 atau yang lebih dikenal dengan nama MP3, adalah salah satu dari pengkodean dalam digital audio dan juga merupakan format kompresi audio yang memiliki sifat "menghilangkan". Istilah menghilangkan yang dimaksud adalah kompresi audio ke dalam format MP3 dengan cara menghilangkan aspek-aspek yang tidak signifikan pada pendengaran manusia untuk mengurangi besarnya file audio.

MP3 sendiri adalah pengembangan dari teknologi sebelumnya sehingga dengan ukuran yang lebih kecil dapat menghasilkan kualitas yang setara dengan kualitas CD (*Corps Diplomatique*). Berikut adalah spesifikasi dari *layer* MP3:



Gambar 2.7 Contoh Spesifikasi File MP3 (Pratowo, 2012)

2.2.7.1 Kompresi Pada File MP3

Kompresi yang dilakukan oleh MP3 seperti yang telah disebutkan di atas, tidak mempertahankan bentuk asli dari sinyal input. Melainkan yang dilakukan adalah menghilangkan suara-suara yang keberadaannya kurang/tidak signifikan bagi sistem pendengaran manusia. Adapun proses yang dilakukan adalah sebagai berikut:

a) Tahap Pertama

Tahapan pertama adalah menggunakan model dari sistem pendengaran manusia dan menentukan bagian yang terdengar bagi sistem pendengaran manusia.

b) Tahap Kedua

Setelah itu tahapan kedua adalah sinyal input yang memiliki domain waktu dibagi menjadi blok-blok dan ditransformasi menjadi domain frekuensi.

c) Tahap Ketiga

Kemudia tahapan ketiga adalah model dari sistem pendengaran manusia dibandingkan dengan sinyal input dan dilakukan proses pemfilteran yang menghasilkan sinyal dengan range frekuensi yang signifikan bagi sistem pendengaran manusia. Proses tersebut adalah pengirisan dua sinyal yaitu sinyal input dan sinyal model dari sistem pendengaran manusia.

d) Tahap Keempat

Tahapan terkahir adalah kuantisasi data, dimana data yang terkumpul setelah pemfilteran akan dikumpulkan menjadi satu keluaran dan dilakukan pengkodean dengan hasil akhir file dengan format MP3.

Proses pengkompresian MP3 dapat menghasilkan keluaran yang hampir setara dengan aslinya disebabkan oleh kelemahan dari sistem pendengaran manusia yang dapat eskploitasi. Berikut adalah beberapa kelemahan dari sistem pendengaran manusia yang digunakan dalam pemodelan:

1. Terdapat beberapa suara yang tidak dapat didengar oleh telinga manusia (diluar jangkauan frekuensi 30-30.000 Hz).
2. Terdapat beberapa suara yang dapat terdengar lebih baik bagi pendengaran manusia dibandingkan suara lainnya.
3. Bila terdapat dua suara yang dikeluarkan secara simultan (terjadi pada waktu yang bersamaan), maka pendengaran manusia akan mendengar yang lebih keras sedangkan yang lebih pelan akan tidak terdengar.

2.3 Metode Observasi

Menurut Esterberg dalam Sugiyono (2013:231) observasi merupakan suatu proses yang tersusun dari berbagai faktor seperti biologis ataupun psikhologis dari pengamatan dan ingatan seseorang.

2.4 Metode Wawancara

Menurut Esterberg dalam Sugiyono (2013:231) wawancara merupakan pertemuan dua orang untuk bertukar informasi dan ide melalui tanya jawab, sehingga dapat dikontruksikan makna dalam suatu topik tertentu. Tanya jawab 'sepihak' berarti bahwa pengumpul data yang aktif bertanya, sermentara pihak yang ditanya aktif memberikan jawaban atau tanggapan. Dari definisi itu, kita juga dapat mengetahuibahwa Tanya jawab dilakukan secara sistematis, telah terencana, dan mengacu pada tujuan penelitian yang dilakukan.

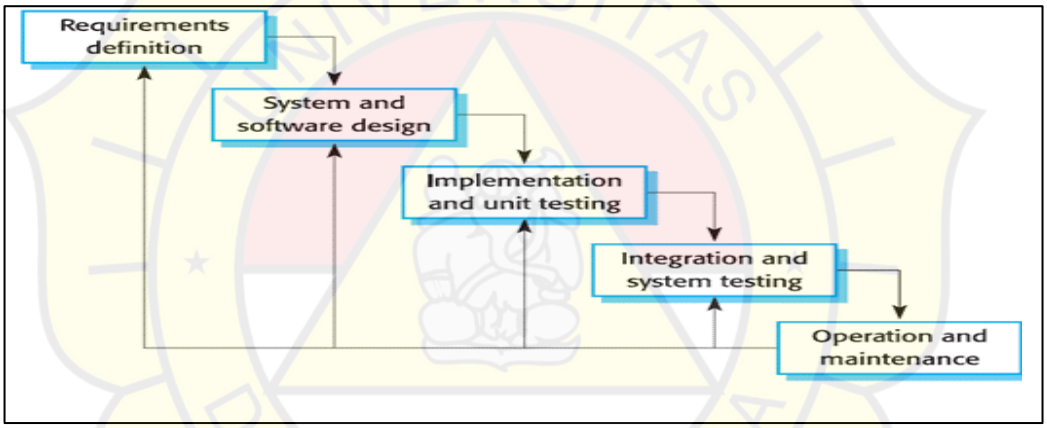
2.5 Metode Studi Pustaka

Menurut Esterberg dalam Sugiyono (2012:231) metode studi pustaka adalah kajian teoritis, referensi serta literatur ilmiah lainnya yang berkaitan dengan budaya, nilai dan norma yang berkembang pada situasi sosial yang diteliti. Secara garis besar, Studi Kepustakaan yaitu mengadakan penelitian dengan cara mempelajari dan membaca literatur-literatur yang ada hubungannya dengan permasalahan yang menjadi obyek penelitian.

2.6 Waterfall

Dalam pengembangan aplikasi Penerapan Metode Steganografi LSB untuk pengiriman kode file *E-budgeting* pada bank XYZ ini, penulis menggunakan metode *waterfall*.

Menurut (Ian Sommerville, 2011) dalam buku *" Software Engineering, 9th Edition"*. Metodologi *waterfall* adalah suatu proses pengembangan perangkat lunak berurutan, di mana kemajuan dipandang sebagai terus mengalir ke bawah (seperti air terjun) melewati fase-fase perencanaan, pemodelan, implementasi (konstruksi), dan pengujian.



Gambar 2.8 Metodologi *Waterfall* (Ian Sommerville, 2011)

Metode Waterfall memiliki tahapan-tahapan sebagai berikut :

1. Requirements analysis and definition Layanan sistem, kendala, dan tujuan ditetapkan oleh hasil konsultasi dengan pengguna yang kemudian didefinisikan secara rinci dan berfungsi sebagai spesifikasi sistem.
2. System and software design Tahapan perancangan sistem mengalokasikan kebutuhan-kebutuhan sistem baik perangkat keras maupun perangkat lunak dengan membentuk arsitektur sistem secara keseluruhan. Perancangan

perangkat lunak melibatkan identifikasi dan penggambaran abstraksi sistem dasar perangkat lunak dan hubungannya.

3. Implementation and unit testing Pada tahap ini, perancangan perangkat lunak direalisasikan sebagai serangkaian program atau unit program. Pengujian melibatkan verifikasi bahwa setiap unit memenuhi spesifikasinya.

4. Integration and system testing Unit-unit individu program atau program digabung dan diuji sebagai sebuah sistem lengkap untuk memastikan apakah sesuai dengan kebutuhan perangkat lunak atau tidak.

Operation and maintenance Biasanya (walaupun tidak selalu), tahapan ini merupakan tahapan yang paling panjang. Sistem dipasang dan digunakan secara nyata. Maintenance.

2.7 Pemodelan Sistem

Menurut (Rumbaugh,Grady Booch,Ivar Jacobson dalam The Unified Modeling Language Reference Manual Second Edition (2004:3), UML (*Unified Modeling Language*) merupakan bahasa pemodelan visual yang umum digunakan untuk menentukan, visualisasi, membangun dan artefak dokumen dari sebuah sistem perangkat lunak. UML pertama kali diperkenalkan Grady Booch dan (Rumbaugh pada tahun 1994, dengan menggabungkan metode pengembangan berbasis objek dengan tujuan menciptakan satu, standard proses untuk pengembangan sistem berbasis objek. Kemudian pada tahun 1995 Ivar Jacobson yang berkontribusi dalam Object-Oriented Software Engineering (OOSE)

bergabung kedalam proyek dan focus untuk membuat standard bahasa pemodelan obyek dengan pendekatan atau metode standar orientasi obyek.

Tabel 2.1 *UML View dan Diagram (Rumbaugh, 2005)*

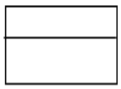
Major Area	View	Diagram	Main Concepts
Dynamic	*State Machine View*	*State Machine Diagram*	*completion transition, do activity,effect, event ,region ,state ,transition ,trigger*
	activity view	*activity diagram*	*Action, activity, control flow, exception, expansion region, fork, join, object node, pin*
	interaction view	*sequence diagram*	*occurrence specification, execution specification, interaction, interaction fragment, interaction operand, lifeline, message, signal*
		communication diagram	*collaboration, guard condition, message, role, sequence number*
physical	*deployment view*	*deployment diagram*	*artifact, dependency, manifestation, node*
model management	*model management view*	*package diagram*	*import, model, package*
	profile	*package diagram*	*constraint, profile, stereotype, tagged value*

2.7.1 *Static View*

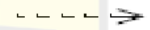
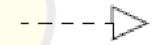
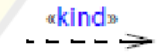
Menurut (Rumbaugh,Grady Booch,Ivar Jacobson dalam The Unified Modeling Language Reference Manual Second Edition (2004:3), Static view adalah pondasi dari UML. Elemen – elemen dari model static view adalah konsep yang sangat berarti pada sebuah aplikasi, termasuk pada konsep nyata, konsep abstrak, konsep implementasi, dan konsep yang berada dalam sistem computer. *Static View*

memperlihatkan struktur obyek. Pada sistem berorientasi obyek, penyatuan struktur data dan perilaku (*behavior*) fitur kedalam satu struktur obyek.

Tabel 2.2 Komponen pembentuk *Class Diagram (Whitten & Bentley ,2007)*

Fungsi	Notasi
Upaya untuk menghindari asosiasi dengan lebih dari dua objek.	
Himpunan dari objek – objek yang berbagi atribut serta operasi yang sama.	
Menggambarkan sebuah sasaran yang merupakan sebuah hasil dari kegiatan keputusan	

Tabel 2.3 Jenis – jenis relasi class diagram dan notasinya (Rumbaugh, 2010)

Nama Relasi	Fungsi	Notasi
Association	Mendeskripsikan koneksi di antara suatu kelas	
Dependency	Hubungan antara dua elemen model	
Generalization	Hubungan lebih spesifik dan deskripsi yang lebih umum ,digunakan untuk jenis deklarasi inheritance dan polymorphic	
Realization	Hubungan antara spesifikasi dan implementasinya	
Usage	Situasi dimana satu elemen membutuh elemen lain untuk berfungsi dengan benar	

2.7.2 *User case View*

Menurut (Rumbaugh,Grady Booch,Ivar Jacobson dalam The Unified Modeling Language Reference Manual Second Edition (2004 : 3), User Case View menggambarkan perilaku dari sistem, subsistem, *class* dan komponen yang tampak bagi pengguna. Bagian – bagian fungsional sistem ke dalam sebuah transaksi berguna bagi *actor* (pengguna sistem). *Actor* adalah diidealkan peran yang

dimainkan oleh seorang eksternal, proses, atau hal yang berinteraksi dengan sistem, subsistem atau *class.* Ada empat macam tipe aktor:

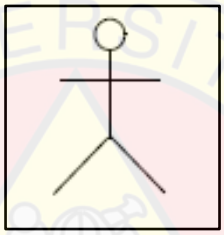
a) Primary business actor (Pelaku bisnis utama)

b) Primary system actor (Pelaku sistem utama)

c) External server actor (Pelaku server eksternal)

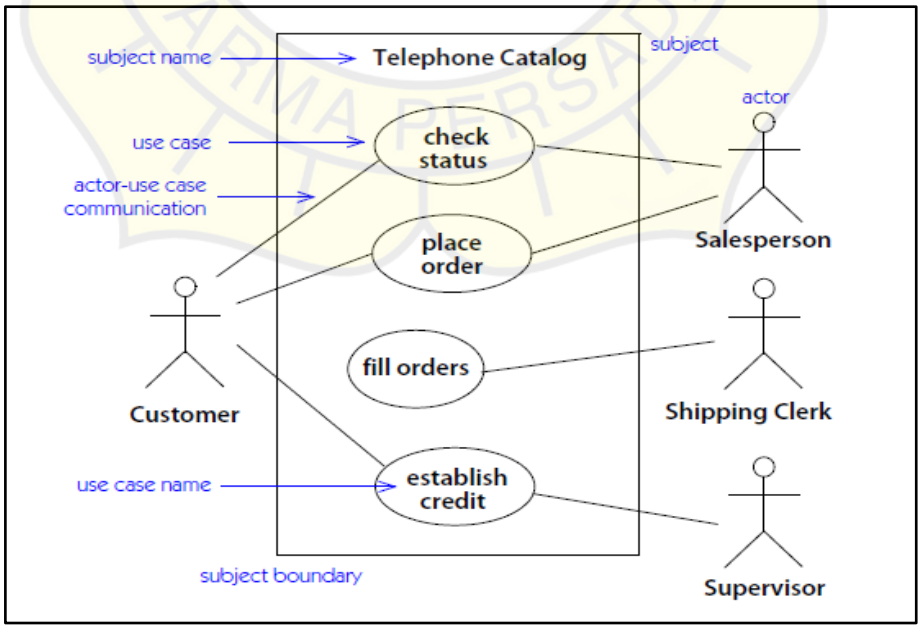
d) External receiving actor (Pelaku penerima eksternal)

Use case Diagram adalah diagram yang menggambarkan interkasi antara sistem dengan sistem eksternal dan pengguna.

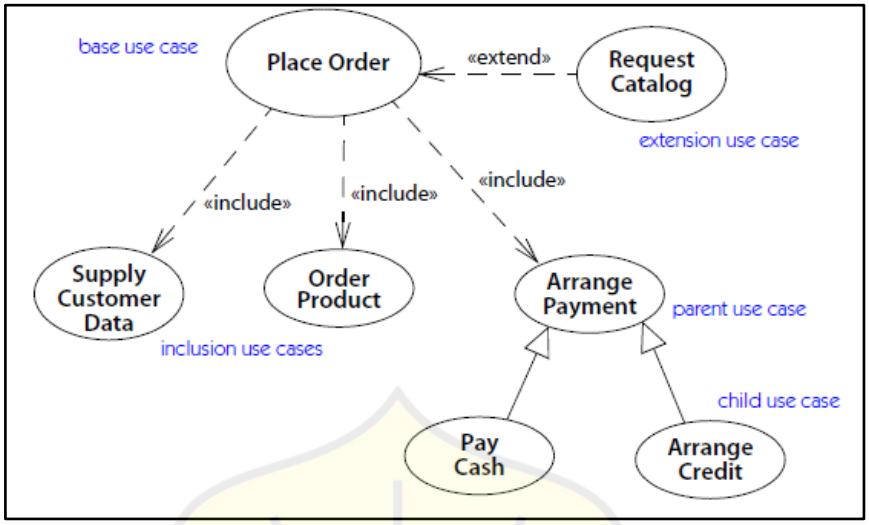


Gambar 2.9 Notasi Actor (Rumbaugh, 2010)

Use case Diagram adalah Diagram yang menggambarkan interaksi antara sistem dengan sistem eksternal dan pengguna.



Gambar 2.10 Contoh *Use Case* Diagram (Rumbaugh, 2010)



Gambar 2.11 Relasi *Use Case* Diagram (Rumbaugh, 2010)

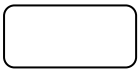
Tabel 2.4 Jenis – jenis relasi class diagram dan notasinya *(*Whitten & Bentley *,2007)*

Nama Relasi	Fungsi	Notasi
Association	Jalur komunikasi di antara actor dan use case yang berpartisipasi di dalamnya	———
Extend	Penyisipan perilaku tambahan ke dalam *use case* utama	«extend» - - - - →
Include	Penyisipan perilaku tambahan ke dalam *use case* utama yang secara eksplisit menjelaskan penyisipan	«include» - - - - →
Use case generalization	Hubungan antara *parent use case* dan *child use case* yang lebih spesifik yang mewarisi dan menambahkan fitur ke dalamnya	——▷

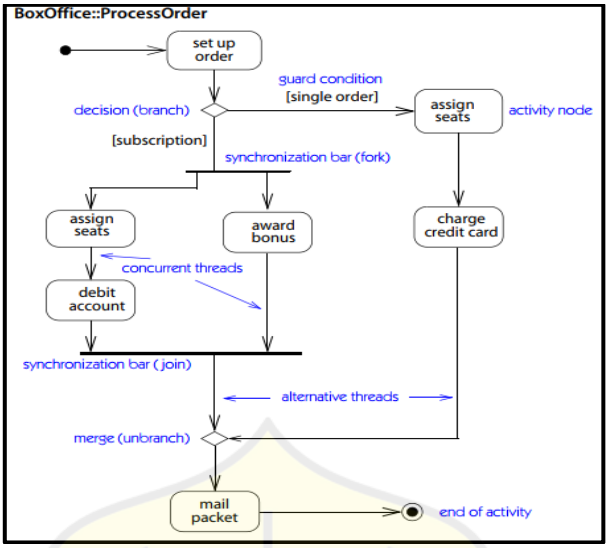
2.7.3 *Activity View*

Menurut (Rumbaugh,Grady Booch,Ivar Jacobson dalam The Unified Modeling opsional) melawati langkah-langkah komputasi. Simpul *Activity* menunjukan.

Tabel 2.5 Komponen Pembentuk Activity Diagram (Whitten & Bentley, 2007)

Nama	Fungsi	Notasi
Initial Node	Menggambarkan awal sebuah proses	
Actions	Menggambarkan sebuah kegiatan atau tugas yang perlu dilakukan	
Flow	Menggambarkan sasaran yang mengawali kegiatan	
Decision	Menggambarkan sebuah kegiatan keputusan	
Merge	Menggambarkan sebuah kegiatan yang dapat muncul secara paralel	
Decision Points	Menggambarkan sebuah sasaran yang merupakan sebuah hasil dari kegiatan keputusan	[ ]
Activity Final	Menggambarkan akhir dari sebuah proses	

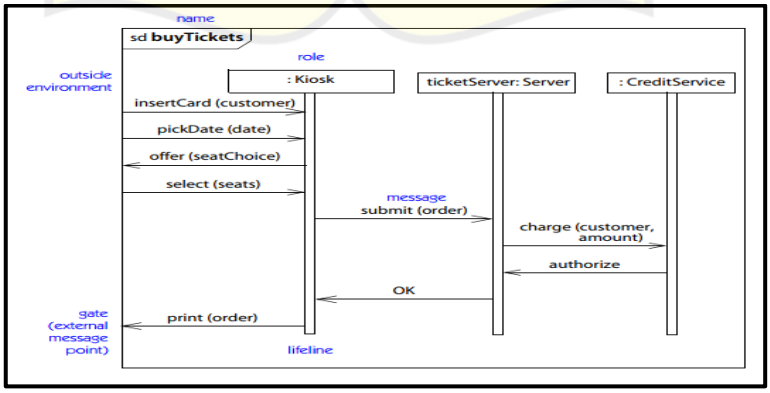
Language Reference Manual Second Edition (2004:3), *Acitivity View* adalah gambar sebuah simpul dan aliran yang menggambarkan aliran kontrol dan data.



Gambar 2.12 Contoh Diagram *Activity* (Rumbaugh, 2010)

2.7.4 *Interaction View*

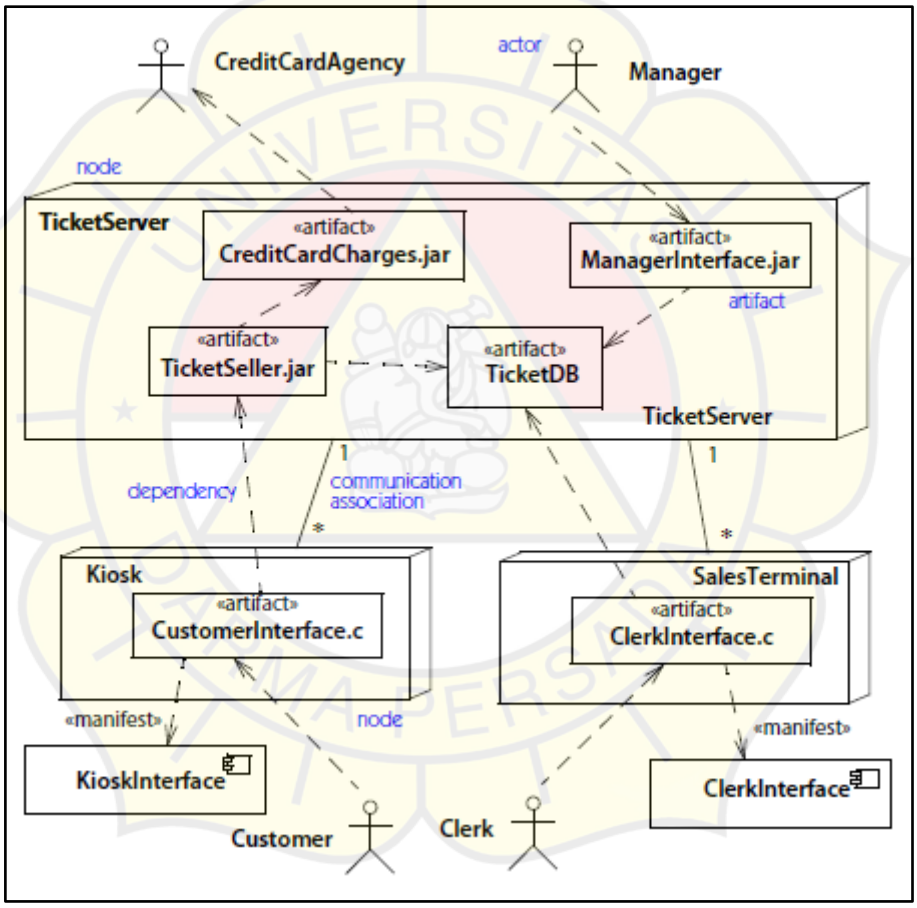
Menurut (Rumbaugh,Grady Booch,Ivar Jacobson dalam The Unified Modeling Language Reference Manual Second Edition (2004 : 3), *Interacion View* membuktikan lebih pandangan holistic dari perilaku dari set obyek. *Interacion View* dimodelkan dengan interaksi pada struktur *classifier*, dan kolaborasi. Interaction adalah satu set pesan yang didalamnya terkandung *structured classifier* atau *collaboration* dimana saling bertukar dengan peran di seluruh konektor. Sequence Diagram adalah diagram yang menggambarkan interaksi antara *actor* dan sistem untuk sebuah scenario *use case*.



Gambar 2.13 Contoh *Sequence* Diagram (Rumbaugh, 2010)

2.7.5 *Deployment View*

Menurut (Rumbaugh,Grady Booch,Ivar Jacobson dalam The Unified Modeling Language Reference Manual Second Edition (2004 : 3), *Deployment View* menunjukkan susunan fisik dari node. *Node* adalah run-time sumber daya komputasi, seperti komputer atau perangkat lain. *Deployment diagram* menunjukan *run-time artifact* pada node. *Artifact* adalah sebuah bagian implementasi seperti sebuah *file.*



Gambar 2.14 Contoh *Deployment* Diagram (Rumbaugh, 2010)