BAB II

TINJAUAN PUSTAKA

II.1. Perancangan

Perancangan adalah suatu tahapan yang memiliki tujuan untuk mendesign sistem baru yang dapat menyelesaikan masalah-masalah yang dihadapin perusahaan yang diperoleh dari pemililihan alternatif sistem yang terbaik. Kegiatan yang dilakukan dalam tahap perancangan ini meliputi perancangan *input*, *output* dan *file*. (AL-Bahra ; 2005 : 39).

Sebagai tahapan setelah siklus pengembangan sistem, pendefinisian dari kebutuhan-kebutuhan fungsional dan persiapan untuk merancang, serta menggambarkan bagaimana suatu sistem dibentuk. (Jogiyanto Hartono ; 2005 : 196).

Berdasarkan penjelasan diatas penulis dapat mengambil kesimpulan bahwa perancangan merupakan kegiatan mendesain sistem baru yang bertujuan untuk menyelesaikan masalah yang dihadapi perusahaan atau suatu kegiatan yang memiliki tujuan untuk mendesain sistem yang baru yang dapat menyelesaikan masalah-masalah yang diadapi perusahaan yang diperoleh dari pemilihan alternatif sistem yang terbaik.

II.2. Aplikasi

Aplikasi berasal dari kata *application* yang artinya penerapan, lamaran, penggunaan. Secara istilah aplikasi program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju. (Sobri ; 2013).

Perangkat lunak aplikasi yaitu perangkat lunak yang digunakan untuk membantu pemakai komputer untuk melaksanakan pekerjaannya. Jika ingin mengembangkan program aplikasi

sendiri, maka untuk menulis program aplikasi tersebut, dibutuhkan suatu bahasa pemrograman, yaitu *language software*, yang dapat berbentuk *assembler*, *compiler* ataupun *interpreter*. Jadi *language software* merupakan bahasanya dan program yang ditulis merupakan program aplikasinya. *Language software* berfungsi agar dapat menulis program dengan bahasa yang lebih mudah, dan akan menterjemahkannya ke dalam bahasa mesin supaya bisa dimengerti oleh komputer. Bila hendak mengembangkan suatu program aplikasi untuk memecahkan permasalahan yang besar dan rumit, maka supaya program aplikasi tersebut dapat berhasil dengan baik, maka dibutuhkan prosedur dan perencanaan yang baik dalam mengembangkannya.

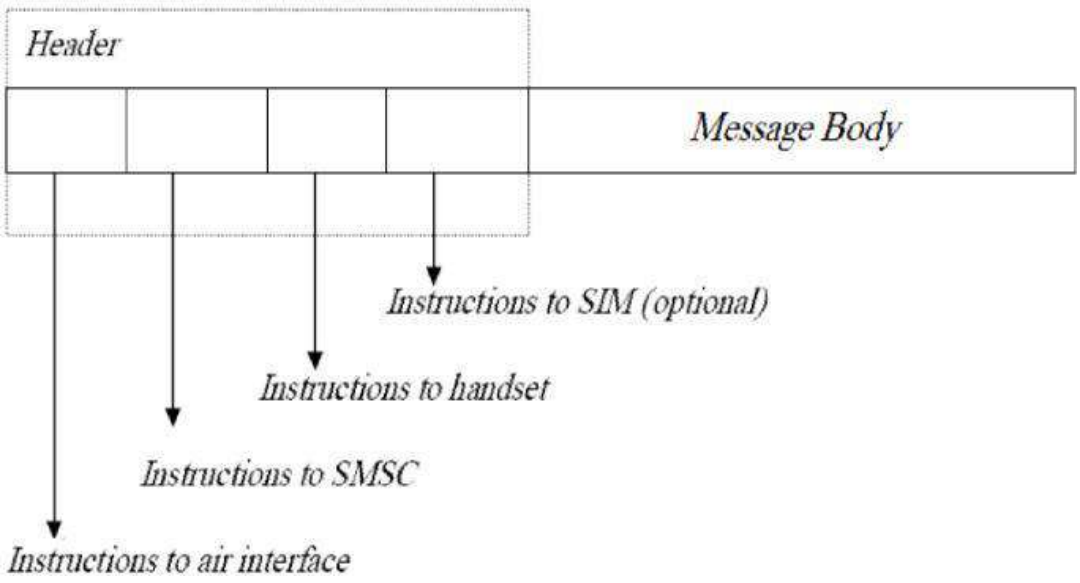
Sekarang, banyak sekali program-program aplikasi yang tersedia dalam bentuk paket-paket program. Ini adalah program-program aplikasi yang sudah ditulis oleh orang lain atau perusahaan-perusahaan perangkat lunak. Beberapa perusahaan perangkat lunak telah memproduksi paket-paket perangkat lunak yang mempunyai reputasi internasional. Program-program paket tersebut dapat diandalkan, dapat memenhi kebutuhan pemakai, dirancang dengan baik, relatip bebas dari kesalahan-kesalahan, *user friendly* (mudah digunakan), mempunyai dokumentasi manual yang memadai, mampu dikembangkan untuk kebutuhan mendatang, dan didukung perkembangannya. Akan tetapi, bila permasalahannya bersifat khusus dan unik, sehingga tidak ada paket-paket program yang sesuai untuk digunakan, maka dengan terpaksa harus mengembangkan program aplikasi itu sendiri. (Jogiyanto Hartono ; 2004).

Dari definisi aplikasi diatas dapat penulis simpulkan pengertian aplikasi adalah suatu program (*software*) yang ditulis atau dirancang untuk menangani suatu masalah.

II.3. SMS (*Short Message Service*)

Short Message Service (SMS) merupakan sebuah layanan yang banyak diaplikasikan pada sistem komunikasi tanpa kabel, memungkinkan dilakukannya pengiriman pesan dalam bentuk teks SMS didukung oleh GSM (*Global System For Mobile Communication*), TDMA (*Time Division Multiple Access*), CDMA (*Code Division Multiple Access*) yang berbasis pada telepon seluler pada saat ini banyak digunakan masyarakat. (Yudi ; 2011).

SMS merupakan salah satu layanan komunikasi yang disediakan oleh telepon seluler untuk berkomunikasi dengan cara mengirimkan pesan singkat dengan cepat dan murah. Adapun struktur pesan SMS sebagai berikut :



Gambar II.1. Struktur Pesan SMS

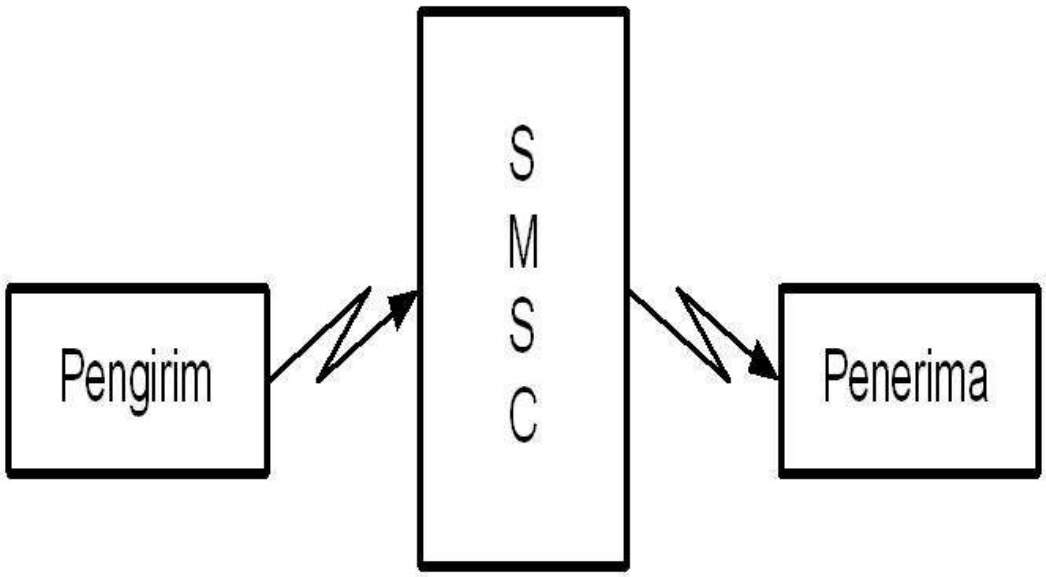
(Sumber : http://developers.sun.com/mobility/midp/articles/sms/)

Pada gambar II.I dapat dilihat struktur pesan SMS terdiri dari bagian header yang berisi instruksi-instruksi yang bekerja dalam jaringan SMS. Sedangkan bagian message body berisi isi

dari pesan yang akan dikirimkan. Pesan SMS memiliki panjang yang berukuran maksimal 160 karakter yang mana setiap karakter memiliki panjang 7 *bit.* (Agus Sarjuni ; 2013)

II.3.1. Cara Kerja SMS

Layanan SMS menggunakan kanal atau jalur teks dalam proses penyampaiannya. Sehingga meskipun sang penerima SMS sedang melakukan kegiatan pembicaraan dalam *handphone*-nya, SMS yang masuk tetap dapat diterima.



Gambar II.2. Skema Sederhana Cara Kerja SMS

(Sumber : http://blog.ub.ac.id/sriningsih/files/2011/06/SMSC.jpg)

Pada gambar II.2 dapat dilihat skema cara kerja SMS Pada saat kita mengirim pesan dari telepon genggam pesan tersebut tidak langsung dikirimkan ke telepon genggam tujuan, akan tetapi dikirim terlebih dahulu ke SMSC baru kemudian pesan tersebut diteruskan ketelepon gengam tujuan. Dengan adanya SMSC ini kita dapat mengetahui status dari pesan yang telah dikirim, apakah telah sampai atau gagal diterima oleh telepon genggam tujuan. Apabila telepon

genggam tujuan dalam keadaan aktif dan dapat menerima SMS yang dikirim ia akan mengambil kembali pesan konfirmasi ke SMSC yang menyatakan bahwa pesan telah dikirim, kemudian SMSC mengirim kembali status tersebut ke si pengirim. Jika telepon genggam tujuan dalam keadaan mati pesan yang kita kirim akan disimpan pada SMSC sampai proses validasi terpenuhi. (Purnomo ; 2007:6)

II.4. Kriptografi

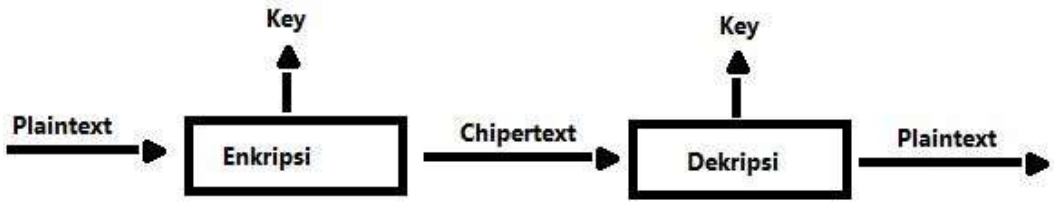
Kriptografi (*cryptography*) berasal dari bahasa Yunani, *crypto* dan graphia. *Crypto* berarti *secret* (rahasia) dan *graphia* berarti *writing* (tulisan). Menurut terminologinya, kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. (Munir ; 2006 : 2).

Enkripsi merupakan proses menyandikan *plaintext* menjadi *ciphertext* dengan menggunakan algoritma tertentu, Proses enkripsi biasanya dilakukan sebelum pesan dikirimkan. Untuk meningkatkan keamanan enkripsi pesan, pada proses enkripsi ditambahkan kunci.

Sedangkan Dekripsi merupakan proses untuk mengubah *cipherteks* kembali menjadi *plainteks* agar pesan dapat dimengerti, Proses dekripsi dilakukan oleh penerima pesan agar pesan yang diterima dapat dimengerti. Untuk proses enkripsi yang menggunakan kunci maka dekripsi harus dilakukan dengan menggunakan kunci.

Kunci/*key* yang digunakan pada proses dekripsi dapat berbeda dengan kunci yang digunakan pada proses enkripsi, disebut juga kriptografi kunci publik. Sebaliknya jika kunci yang digunakan sama, disebut juga kriptografi kunci simetri. (Agus Sarjuni ; 2013)

Adapun proses cara kerja kriptografi dapat dilihat pada gambar berikut:



Gambar II.3. Proses Enkripsi dan Dekripsi

(Sumber : http://www.rumus.xyz/2016/03/rumus-dasar-kriptografi-enkripsi.html)

II.5. Analisa Penerapan Algoritma RC6

RC6 adalah algoritma blok kode yang sangat aman, padat, sederhana dan menawarkan performansi yang sangat bagus dan fleksibel, dikembangkan dari RC5 oleh Ronald Linn Rivest, Ray Sidney, Matt JB.Robshaw dan Yiquin Yin dari RSA security, Inc. Pada tahun 1998. Seperti halnya RC5 parameter daru algoritma ini adalah ukurun blok, ukuran kunci eksternal dan jumlah putaran yang bervariasi dengan batasan sama seperti pada RC5 (Dony Ariyus, 2008). Algoritma RC6 juga lebih unggul dalam kecepatan proses enkripsi dibanding algoritma lainnya seperti RC4 dan Blowfish. (Yudi Prayudi ; 2005)

Algoritma RC6 merupakan algoritma sederhana, fungsi yang digunakan merupakan fungsi yang sederhana dan hanya mengandalkan prinsip teknik cipher berulang (iterated cipher) untuk keamanan. Tampilan hasil enkripsi dan data hasil enkripsi yang diterima setiap karakternya memiliki panjang 8 bit, sedangkan sebagian telepon seluler hanya dapat menampilkan karakter dengan panjang 7 bit. Dengan demikian dalam penerapan algoritma RC6 pada SMS karakter-karakter yang akan dienkripsi diubah kedalam nilai ASCII, dimana nilai karakter dalam table ASCII ditambah dalam karakter special adalah 0 sampai 244, artinya satu

karater ASCII akan diwakili oleh 8 bit, dimana 28 = 256. Sehingga, dalam 1 blok plainteks (32 bit) akan menyimpan 4 karakter dan setiap kali iterasi, maka akan diambil 16 karakter plainteks. (Rionald Ricardo Mangundap, Wiwin Agus Kristiana ; 2015)

Apabila panjang plainteks atau panjang kunci kurang dari 16 karakter, maka akan dilakukan padding, yaitu dengan menambahkan bit ”0” (nol) di akhir teks, sehingga panjang teks mencukupi 16 karakter. Layar pada sebagagian besar telepon selular hanya dapat menampilkan karakter dengan panjang 7 bit dan pesan yang telah terenkripsi akan berbentuk binary, sehingga layar tidak akan menampilkan dengan semestinya. Oleh karena itu, pada aplikasi yang akan dibangun, untuk menampilkan pesan yang telah terenkripsi, ditambahkan informasi karakter yang terdapat pesan tersebut dengan format heksadesimal agar dapat ditampilkan dilayar dan informasinya lebih terbaca. (Rionald Ricardo Mangundap, Wiwin Agus Kristiana ; 2015)

Algoritma RC6 yang akan digunakan dalam aplikasi enkripsi SMS yang akan dibangun dengan W sebesar 32 bit, R sebsar 20 kali putaran dan panjang kunci beragam lebih dari 1 karakter (8 bit). Langkah-langkah algoritma RC6 dalam pelaksanaan Proyek Madya ini akan dikelompokkan kedalam beberapa bagian, yaitu:

1. Pembangkit Subkunci

Kunci dari pengguna ini akan dimasukkan oleh pengguna pada saat akan melakukan proses enkripsi dan dekripsi. Kunci ini memiliki tipe data string dan memiliki panjang 16 byte (16 karakter).

2. Baca masukkan untuk proses enkripsi

Yang dilakukan pada tahapan ini adalah membaca teks yang menjadi masukan pada proses enkripsi, yaitu field dari aplikasi enkripsi SMS. Pada proses enkripsi pesan, field-nya adalah isi pesan.

3. Enkripsi meliputi whitening awal, iterasi dan whitening akhir.

4. Baca masukkan untuk Proses Dekripsi

Yang dilakukan pada tahapan ini adalah membaca teks yang menjadi masukkan pada proses dekripsi, yaitu record dari hasil pesan yang telah dienkripsi pada pengirim dan menjadi field pesan pada penerima.

5. Dekripsi merupakan kebalikan dari proses enkripsi. (Rionald Ricardo Mangundap, Wiwin Agus Kristiana ; 2015)

1.Algoritma Pembangkit Sub Kunci

```
Kamus
Type Word32 : 32 bit (tipe data 32 bit)
Kunci : String { kunci yang dimasukkan oleh pengguna}
I, j, c, s, v : integer
A :integer
B : Integer
S : array [0..43] of word 32
L : array [0..43] of word 32
Function
ROTL (X:Word32; y: integer) – Word 32 {fungsi untuk merotasi bit sebanyak variable kedua}
Algoritma
Input (kunci)
S(0) – b7e15163
For I – 1 to 43 do
S[i] – s[i-1] + 9e3779b9
Endfor
A – B – I – j – 0
V – 44
If {c>v} then
v – c
v v*3
For s – 1 to v do
A – S[i] – ROTL ((S[i] + A + B). 3
S – L[j] = ROTL (L[j] + A + B, A + B)
```

```
I – (i+1) mod 44
J – (j+1) mod c
Endfor
```

2. Algoritma Baca File Masukan Proses Enkripsi

```
Prosedur Baca_Masukan_proses_Enkripsi
{Input : Field masukan belum dibaca}
{Output : Field masukan dibaca per 16 karakter dan ditampung dalam buffer. Pada proses ini pesan, filed nya adalah isi pesan }
Kamus
Field_masukan ; string
Buff : array [0..15] of char
i : integer
algoritma
Input (field_masukan)
i – 0
while (I <=15) and not (EOF) do
Read (field_masukan, Buff [i])
Endwhile
```

3. Prosedur Whitening awal

```
{input : blok kedua dan keempat belum dijumlahkan dengan sub kunci}
{output : blok kedua dan keempat yang telah dijumlahkan dengan sub kunci}
Kamus
Type word32 : 32 bit (tipe data sebesar 32 bit)
I : word32 array [0..3] (blok enkripsi/planteks)
E : Array [0..43] of word 32 (sub kunci)
Algoritma
K[1] – X[1] + S[0]
K[3] – X[3] + S[1]
```

3.1 Algoritma Iterasi

```
Prosedur Iterasi
{input : keempat blok setelah whitening awal belum diproses}
{Output : keempat blok yang telah diproses dan saling dipertukarkan}
Kamus
Type word32 : 32 bit {tipe data sebesar 32 bit}
X : word array [0..3] {blok enkripsi/planteks}
Function
ROTL(X : Word32; Y : integer} – word32
{merotasi bit kekiri sebanyak variable kedua}
```

```
Temp : word32
U, t : word32
I : integer
Algoritma
For I – 1 to 20 do
t – ROTL ( (X[1]*(2*X[1]+1)), 5)
u – ROTL ( (X[3]*(2*X[3]+1)), 5)
X[0] – (ROTL ( (X[0] XOR t), u)) + S[2*i}
X[2] – (ROTL ( (X[2] XOR u), t)) + S[2*I + 1]
Temp – X[0]
X[1] – X[1]
X[2] – X[2]
X[3] – Temp
End for
```

3.3 Algoritma Whitening Akhir

```
Prosedur Whitening_akhir
{input : blok pertama dan ketiga belum dijumlahkan dengan sub kunci}
Output : blok pertama dan ketiga yang telah dijumlahkan dengan sub kunci}
Kamus
Type word32 : 32 bit (tipe data sebesar 32 bit)
X : word32 array [0..3] blok enkripsi/planteks
S : Array [0..43] of word 32 (sub kunci)
Algoritma
X[0] – X[0] + S[42]
X[2] – X[0] + S[43]
```

4. Algoritma Baca File Masukan Proses Dekripsi

```
Prosedur Baca_File_Masukan_Proses_Dekripsi
{input : Field masukan berupa chiperteks}
(output : Field pada isi pesan yang berupa chiperteks dibaca per 16 karakter dan ditampung dalam buffer}
Kamus
Field_masukan : string
Buff : array [0..15]
i : integer
```

```
Algoritma
Input (field_masukan)
i – 0
while (i <= 15) and not (field_masukan.EOF) do
Read (isi_kolom, Buff [i])
Endwhile
```

5. Algoritma Dekripsi

```
Prosedur Dekripsi
{input : keempat blok belum diproses}
{output : keempat blok yang telah diproses dans aling dipertukarkan}
Kamus
Type word32 : 32 bit {tipe data sebesar 32 bit}
X : word32 array [0..3] {blok dekripsi/ciperteks}
Function
ROTL (X:word32; Y:integer) – word32 {merotasi bit kekiri sebanyak variable kedua}
Temp : word32
u, t :word32
I : integer
Algoritma
X[2] – X[2] – S[43]
X[0] – X[0] – S[42]
For I – 20 down to 1 do
Temp – X[3]
X[3] – X[2]
X[2] – X[1]
X[1] – X[0]
u – ROTL ( (X[3]*(2*X(3)+1)), 5)
t – ROTL ( (X[1]*(2*X[1]+1}}, 5)
X[2] – (ROTL (X[2] – S(2*1+1]), t) XOR u)
X[0] – (ROTL (X[0] – S[2*i]), u) XOR t)
End for
X[3] – X[3] – S[1]
X[1] – X[1] – S[0]
```

(Rionald Ricardo Mangundap, Wiwin Agus Kristiana ; 2015)

Tabel Kunci Sbox RC6 - 2011			
Sbox	Hexa	Decimal	Binary
0	$B7E15163	3.084.996.963	1011 0111 1110 0001 0101 0001 0110 0011
1	$5618CB1C	1.444.465.436	0101 0110 0001 1000 1100 1011 0001 1100
2	$F45044D5	4.098.901.205	1111 0100 0101 0000 0100 0100 1101 0101
3	$9287BE8E	2.458.369.678	1001 0010 1000 0111 1011 1110 1000 1110
4	$30BF3847	817.838.151	0011 0000 1011 1111 0011 1000 0100 0111
5	$CEF6B200	3.472.273.920	1100 1110 1111 0110 1011 0010 0000 0000
6	$6D2E2BB9	1.831.742.393	0110 1101 0010 1110 0010 1011 1011 1001
7	$0B65A572	191.210.866	0000 1011 0110 0101 1010 0101 0111 0010
8	$A99D1F2B	2.845.646.635	1010 1001 1001 1101 0001 1111 0010 1011
9	$47D498E4	1.205.115.108	0100 0111 1101 0100 1001 1000 1110 0100
10	$E60C129D	3.859.550.877	1110 0110 0000 1100 0001 0010 1001 1101
11	$84438C56	2.219.019.350	1000 0100 0100 0011 1000 1100 0101 0110
12	$227B060F	578.487.823	0010 0010 0111 1011 0000 0110 0000 1111
13	$C0B27FC8	3.232.923.592	1100 0000 1011 0010 0111 1111 1100 1000
14	$5EE9F981	1.592.392.065	0101 1110 1110 1001 1111 1001 1000 0001
15	$FD21733A	4.246.827.834	1111 1101 0010 0001 0111 0011 0011 1010
16	$9B58ECF3	2.606.296.307	1001 1011 0101 1000 1110 1100 1111 0011
17	$399066AC	965.764.780	0011 1001 1001 0000 0110 0110 1010 1100
18	$D7C7E065	3.620.200.549	1101 0111 1100 0111 1110 0000 0110 0101
19	$75FF5A1E	1.979.669.022	0111 0101 1111 1111 0101 1010 0001 1110
20	$1436D3D7	339.137.495	0001 0100 0011 0110 1101 0011 1101 0111
21	$B26E4D90	2.993.573.264	1011 0010 0110 1110 0100 1101 1001 0000
22	$50A5C749	1.353.041.737	0101 0000 1010 0101 1100 0111 0100 1001
23	$EEDD4102	4.007.477.506	1110 1110 1101 1101 0100 0001 0000 0010
24	$8D14BABB	2.366.945.979	1000 1101 0001 0100 1011 1010 1011 1011
25	$2B4C3474	726.414.452	0010 1011 0100 1100 0011 0100 0111 0100
26	$C983AE2D	3.380.850.221	1100 1001 1000 0011 1010 1110 0010 1101
27	$67BB27E6	1.740.318.694	0110 0111 1011 1011 0010 0111 1110 0110
28	$05F2A19F	99.787.167	0000 0101 1111 0010 1010 0001 1001 1111
29	$A42A1B58	2.754.222.936	1010 0100 0010 1010 0001 1011 0101 1000
30	$42619511	1.113.691.409	0100 0010 0110 0001 1001 0101 0001 0001
31	$E0990ECA	3.768.127.178	1110 0000 1001 1001 0000 1110 1100 1010
32	$7ED08883	2.127.595.651	0111 1110 1101 0000 1000 1000 1000 0011
33	$1D08023C	487.064.124	0001 1101 0000 1000 0000 0010 0011 1100
34	$BB3F7BF5	3.141.499.893	1011 1011 0011 1111 0111 1011 1111 0101
35	$5976F5AE	1.500.968.366	0101 1001 0111 0110 1111 0101 1010 1110
36	$F7AE6F67	4.155.404.135	1111 0111 1010 1110 0110 1111 0110 0111
37	$95E5E920	2.514.872.608	1001 0101 1110 0101 1110 1001 0010 0000
38	$341D62D9	874.341.081	0011 0100 0001 1101 0110 0010 1101 1001
39	$D254DC92	3.528.776.850	1101 0010 0101 0100 1101 1100 1001 0010
40	$708C564B	1.888.245.323	0111 0000 1000 1100 0101 0110 0100 1011
41	$0EC3D004	247.713.796	0000 1110 1100 0011 1101 0000 0000 0100
42	$ACFB49BD	2.902.149.565	1010 1100 1111 1011 0100 1001 1011 1101
43	$4B32C376	1.261.618.038	0100 1011 0011 0010 1100 0011 0111 0110

Gambar II.4. Proses Perhitungan Sbox

(Sumber : http://catatanrimbun.blogspot.co.id/2012/12/algoritma-rivest-code-6-rc6.html)

II.6. Bahasa Pemrograman Java

Bahasa pemrograman Java muncul pada tahun 1991 ketika perusahaan *Sun Microsystem* memulai *Green Project*, yakni proyek penelitian untuk membuat bahasa yang akan digunakan pada *chip-chip embedded* untuk *device intelligent customer electronic*. Bahasa tersebut haruslah bersifat *multiplatform*, tidak tergantung kepada vendor yang memanufaktur *chip* tersebut. Karena pada awalnya ditujukan untuk pemrograman *device* kecil. Java memiliki karakteristik berukuran kecil, efisien dan protable untuk berbagai *hardware*.

Java 2 Enterprise Edition (J2EE) adalah kelompok dari beberapa API dari *Java* dan teknologi selain *Java*. J2EE digunakan untuk aplikasi berbasi web, aplikasi sistem tersebar dengan beraneka ragam klien dengan kompleksitas yang tinggi.

1. *Java 2 Second Edition* (J2SE) adalah inti dari bahasa pemrograman Java dan digunakan untuk membuat aplikasi dikomputer desktop. *Java Development Kit* (JDK) adalah salah satu *tool J2SE* untuk mengkompilasi program Java dan JRE.
2. *Java 2 Micro Edition* (J2ME) merupakan bagian dari J2SE, dan salah satu aplikasinya banyak dipakai adalah untuk *wireless device* atau *mobile device*. (Agus Sarjuni ; 2013)

II.7. Android

Menurut Burnette (2010 ; 18), Android adalah sistem operasi untuk *mobile* atau telepon selular yang berbasis *Linux*. Android menyediakan *platform* terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam perangkat *Mobile*. Awalnya *Google Inc* (*Google Incoporated*). membeli Android Inc, pendatang baru yang

membuat perangkat lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan perangkat keras, perangkat lunak, dan telekomunikasi, termasuk *Google, HTC, Intel, Motorola, Qualcomm, T-Mobile,* dan *Nvdia*. Pada saat perilisan Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standart terbuka pada perangkat seluler. Di lain pihak, Google merilis kode-kode Android dibawah lisensi *Apache*, sebuah lisensi perangkat lunak dan standart terbuka perangkat seluler.

II.7.1. Versi Android Operasi Sistem

1. Android Versi 1.1

Google merilis Android versi 1.1 pada tanggal 9 Maret 2009. Apa saja fiturnya? Android versi ini dilengkapi dengan tampilan yang lebih cantik pada aplikasi, penambahan fitur pada jam alarm, *voice search* (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email. (Agus Sarjuni ; 2013)

Gambar II.5. Android versi 1.1

(Sumber : http://kundang.weblog.esaunggul.ac.id/wp-content/uploads/sites/99/2013/09/Android-1.1.jpg)

2. Android Versi 1.5 (Cupcake)

Google kembali merilis perangkat smartphone menggunakan Android dan SDK (*Software Development Kit*) versi 1.5 (Cupcake) pada pertengahan Mei 2009. Ada beberapa pembaharuan di sini, di antaranya penambahan beberapa fitur seperti kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke *Youtube* dan gambar ke *Picasa* secara langsung dari telepon. Ada juga Bluetooth AD2P yang memungkinkan hubungan secara otomatis ke headset Bluetooth. Ditambahkan juga adanya animasi layar, serta *keyboard* pada layar yang dapat disesuaikan dengan sistem. (Agus Sarjuni ; 2013)

Gambar II.6. Android Cupcake

(Sumber : http://www.hongkiat.com/blog/android-evolution/)

3. Android Versi 1.6 (Donut)

Versi ini diberi nama Donut dan dirilis pada bulan September 2009 di tahun yang sama dengan versi 1.5. Di sini ada fitur baru yang antara lain proses pencarian yang lebih baik dibanding sebelumnya, adanya indikator baterai dan control applet VPN.

Fitur lainnya adalah galeri yang memungkinkan penguna memilih foto yang akan dihapus. Fitur kamera, *camcorder*, dan galeri juga diintegrasikan. Dukungan ke standar telekomunikasi yang lain, seperti CDMA/EVDO, 802.1x, VPN, *Gesture*, dan *Text-to-speech engine.* Juga ada kemampuan dial kontak, dan teknologi *text-to-speech change speech* (tidak tersedia pada semua ponsel), serta dukungan terhadap resolusi VWGA. (Agus Sarjuni ; 2013)

Gambar II.7. Android Donut

(Sumber : http://www.hongkiat.com/blog/android-evolution/)

4. Android Versi 2.0/2.1 (Eclair)

Pada tanggal 3 Desember 2009, Open Handset Alliance kembali meluncurkan ponsel Android dengan versi 2.0/2.1 (Eclair), perubahan yang dilakukan adalah optimasi hardware, adanya Google Maps 3.1.2, perubahan UI dengan browser baru dan dukungan penuh terhadap standar HTML5, daftar kontak diperbarui serta adanya dukungan flash untuk kamera 3,2 MP, digital Zoom, dan Bluetooh II.7. (Agus Sarjuni ; 2013)

Gambar II.8. Android Eclair

(Sumber : http://www.hongkiat.com/blog/android-evolution/)

5. Android Versi 2.2 (Froyo (Frozen Yoghurt))

Pada tanggal 20 Mei 2010, Android versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan dibandingkan versi-versi sebelumnya antar lain dukungan *Adobe Flash* 10.1, memiliki kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, integrasi *JavaScript engine* versi 8 yang dipakai *Google Chrome* sehingga mempercepat kemampuan *rendering browser*. Ada juga pemasangan aplikasi dalam SD Card, kemampuan *Wifi Hotspot portable*, dan kemampuan auto *update* dalam aplikasi Android Market. (Agus Sarjuni ; 2013)

Gambar II.9. Android Froyo

(Sumber : http://www.hongkiat.com/blog/android-evolution/)

6. Android Versi 2.3 (Gingerbread)

Pada tanggal 6 Desember 2010, Android versi 2.3 (*Gingerbread*) diluncurkan. Perubahan-perubahan yang ada di Android versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antarmuka (*User Interface*) didesain ulang, dukungan format video VP8 dan WebM, ada juga efek audio baru (*reverb, equalization, headphone virtualization,* dan *bass boots*), serta dukungan kemampuan *Near Field Communication* (NFC) di perangkat ponsel, serta dukungan jumlah kamera yang lebih dari satu. (Agus Sarjuni ; 2013)

Gambar II.10. Android Gingerbread

(Sumber : http://www.hongkiat.com/blog/android-evolution/)

7. Android Versi 3.0/3.1 (Honeycomb)

Android Honeycomb dirancang khusus untuk bisa mengakomodasikan tablet. Android versi ini mendukung ukuran layar yang lebih besar. *User Interface* pada Honeycomb juga

berbeda karena sudah didesain untuk tablet. Honeycomb mendukung multi prosesor dan juga akselerasi perangkat keras (*hardware*) untuk grafis. Tablet pertama yang dibuat dengan menjalankan Honeycomb adalah Motorola Xoom. (Agus Sarjuni ; 2013)

Gambar II.11. Android Honeycomb

(Sumber : http://www.hongkiat.com/blog/android-evolution/)

8. Android Versi 4.0 (ICS: Ice Cream Sandwich)

Versi Android ini diumumkan pada tanggal 19 oktober 2011, dengan fitur baru termasuk bisa membuka kunci dengan pengenalan wajah, jaringan data control pemantauan penggunaan, integrasi dengan *social network*, perangkat tambahan fotografi, bisa mencari email secara *offline*, dan berbagi informasi dengan menggunakan *Near Field Communication* (NCF). (Agus Sarjuni : 2013)

Gambar II.12. Android ICS

(Sumber : http://www.hongkiat.com/blog/android-evolution/)

9. Android Versi 4.1 (Jellybean)

Versi Android ini diumumkan pada bulan Juli 2012. Android ICS sudah cukup mempesona dengan tingkat responsifitas yang lebih baik, namun Jelly Bean menawarkan kenyamanan yang lebih berkat *Project Butter*. Google mengklaim ada 3 hal yang membuat Project Butter tampil mempesona, *Vsync* untuk anti *flickr*, lalu *Triple Buffering* untuk mengoptimalkan *OpenGL*, kemudian optimalisasi pada prosesor untuk meningkatkan responsifitas ponsel. Jadi bukan cuma sentuhan saja yang direspon dengan cepat, Jelly Bean juga membuat perpindahan aplikasi semakin halus. (Agus Sarjuni : 2013)

Gambar II.13. Android Jelly Bean

(Sumber : http://www.hongkiat.com/blog/android-evolution/)

II.7.2. Android SDK (*Software Dvelopment Kit*)

Android SDK adalah tools API (*Application Programming Interface*) yang diperlukan untuk memulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemrograman Java. Saat ini disediakan Android SDK sebagai alat bantu dan API untuk mulai mengembangkan aplikasi *platform* Android menggunakan pemrograman Java. (Agus Sarjuni : 2013)

II.7.3. *Eclipse*

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua *platform* (*platform-independent*). *Eclipse* pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari *eclipse* yang membuatnya popular adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan plug-in. (Agus Sarjuni ; 2013)

II.7.4. ADT (*Android Development Tools*)

ADT adalah plug-in yang membuat *Eclipse* dapat membuat project berbasi Android. ADT harus di-instal, karena sebagai penghubung antara Android SDK dengan IDE yang akan dugunakan sebagai tempat koding aplikasi Android nantinya. (Agus Sarjuni ; 2013)

II.7.5. *Intent*

Intent merupakan bagian utama dari aplikasi, yaitu *activities, services*. Dan *broadcast receivers* yang diaktifkan melalui pesan. Intents melayani mekanisme untuk melewatkan pesan antar aplikasi maupun dalam aplikasi itu sendiri. *Intents* juga dapat digunakan untuk memulai activity. Dengan *Intents* juga dapat menyiarkan *action* yang diinginkan misalnya menelepon, melalui sistem ke aplikasi untuk menanganinya. (Agus Sarjuni ; 2013)

II.8. UML (*Unifield Modeling Language*)

Unified Modeling Language adalah satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarakan sebuah sistem software yang terkait dengan objek. Pengertian lain dari UML adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasi, menspesifikasikan, membangun dan pendokumentasian dari sebuah sistem pengembangan software bebrbasis OO (Object-Oriented). UML juga memberikan standar penulisan sebuah sistem blue print, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen-komponen yang diperlukan dalam sistem software.

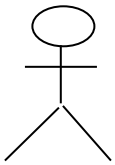
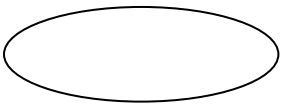
Secara resmi bahasa UML dimulai pada bulan oktotober 1994, ketika Rumbbaugh Booch untuk membuat sebuah project pendekatan metode yang uniform atau seragam dari masing-masing metoda mereka. Saat itu baru dikembangkan draft metoda UML version o,8 dan diselesaikan serta di realese pada bulan oktober 1995.

1. Diagram Use Case

Diagram *Use Case* mengambarkan apa saja aktivitas yang dilakukan oleh suatu sistem dari sudut pandang pengamatan luar yang menjadi persoalan itu apa yang dilakukan bukan bagaimana melakukanya. Menurutu Whittten, (2004) usecase diagram adalah diagram yang mengggambarkan interaksi antara sistem dengan sistem eksternal dan pengguna. Dengan kata lain, secara grafis menggambarkan siapa yang akan menggunakan sistem dan dengan cara apa pengguna mengharapkan untuk berinteraksi dengan sistem.

Adapun simbol-simbol yang digunakan pada saat pembuatan *use case diagram* diperlihatkan pada table.II.1 berikut :

Tabel II.1 Simbol *Use Case* Diagram

Simbol	Keterangan
	Aktor : Seseorang atau sesuatu yang berinteraksi dengan system yang sedang dikembangkan.
	*Use case*: Peringkat tertinggi dari fungsionalitas yang dimilki system.

	*Association*: adalah relasi antara *actor* Dan *use case.*
	Generalisasi : untuk memperlihatkan struktur pewaris yang terjadi.

(Sumber :Verdi Yasin M.Kom.,M.Kom ; 2012 : 270)

2. Diagram Class

Diagram class memberikan pandangan secara luas dari suatu sistem dengan menunjukkan kelas-kelasnya dan hubungan mereka. Diagram class bersifat statis; mengggambarakan hubungan apa yang terjadi jika mereka berhubungan. Diagram class mempunyai 3 macam relationships (hubungan), sebagai berikut :

a. ***Association*** suatu hubungan antara bagian dari dua kelas. Terjadi association antara dua kelas jika salah satu bagian dari kelas mengetahui yang lainya dalm melakukan suatu kegiatan. Didalam diagram, sebuah *association* adalah penghubung yang menghubungkan dua kelas/

b. ***Aggregation*** suatu association dimana salah satu kelasnya merupakan bagian dari suatu kumpulan. Aggregation memiliki titik pusat yang mencakup keseluruhan bagian. Sebagai contoh : Order detail merupakan kumpulan dari order.

c. ***Generalization*** suatu hubungan turunan dengan mengansumsikan suatu kelas merupakan suatu superclass (kelas super) dari kelas lain. Generalization memiliki tingkatan yang berpusat pada supercalss. Contoh : payment adalah superclass dari cash, check, dan kredit.

3. Diagram sequence

Diagran class dan *diagram object* merupakan suatu gambaran model statis. Namun ada juga yang bersifat dinamis, seperti *diagram interaction*. *Diagram sequence* merupakan salah satu diagram interaction yang menjelaskan bagaimana suatu operasi itu dilakukan *message* (pesan) apa yang dikirim dan kapan pelaksanaanya. Diagram ini diatur berdasarkan waktu. Obyek-obyek yang berkaitan dengan proses berjalanya operasi diurutkan dari kiri ke kanan bersasarkan waktu terjadinya dalam pesan yang terurut.

4. **Diagram *Activity***

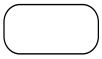
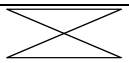
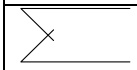
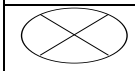
Diagram *Activity* menggambarkan berbagai alir aktivitas dalam sistem yang sedang diarancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi dan bagaimana mereka berakhir. *Activity* diagram juga dapat menggambarkan proses pararel yang mungkin terjadi pada beberapa eksekusi. *Activity* diagram merupakan *state* diagram khusus, dimana sebagian besar state adalah action dan sebagian besar transisi ditrigger oleh selesainya state sebelumnya (*internal processing*) oleh karena itu *activity* diagram tidak menggambarkan behavior internal sebuah sistem (dan interaksi antar sub sistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana actor menggunakan sistem untuk melakukan aktivitas. (Nazruddin Safaat H ; 2015 : 177-181)

Adapun simbol-simbol yang digunakan pada saat pembuatan *activity diagram* diperlihatkan pada table.II.2 berikut :

Tabel II.2 Simbol Activity Diagram

Simbol	Keterangan
●	Titik awal

	Titik akhir
	Activity
	Pilihan untuk pengambilan keputusan
	Fork; untuk menunjukkan kegiatan yang dilakukan secara parallel
	Rake; menunjukkan adanya dekomposisi
	Tanda waktu
	Tanda penerimaan
	Aliran akhir (Flow Final)

(Sumber :Verdi Yasin M.Kom.,M.Kom ; 2012 : 271)