BAB II
TINJAUAN PUSTAKA

2.1 Kriptografi

Kriptografi berasal dari bahasa Yunani "*cryptos*" artinya rahasia, dan "*graphein*" artinya tulisan. Jadi, kriptografi berarti tulisan rahasia. Schneier (1996) menyatakan "*cryptography is the art and science of keeping message secure*''. Kriptografi adalah ilmu sekaligus seni untuk menjaga keamanan pesan (*message*) dengan cara menyandikan (*encrypt*) ke dalam bentuk yang tidak dapat dimengerti lagi makna pesannya. Pada kriptografi terdapat dua proses utama yakni *encoding* dan *decoding*. Proses *encoding* dan *decoding* diatur oleh satu atau lebih kunci kriptografi (Wirdasari, 2008).

Kriptografi klasik yang menitikberatkan kekuatan pada kerahasiaan algoritma, yang berarti apabila algoritma yang digunakan telah diketahui. Maka pesan dapat diketahui oleh siapa saja yang memahami algoritmanya. Kriptografi *modern* lebih menitikberatkan pada kerahasiaan kunci yang dipakai pada algoritma tersebut. Sehingga algoritma tersebut dapat saja disebarkan kepada masyarakat tanpa harus khawatir kehilangan kerahasiaan bagi pemakainya (Wasino dkk, 2012).

2.2 Steganografi

Kata steganografi berasal dari bahasa Yunani "*steganos"*, yang artinya tersembunyi atau terselubung dan "*graphein"* artinya menulis. Steganografi merupakan suatu ilmu, teknik, dan seni menyembunyikan data rahasia pada sebuah media sehingga keberadaan data rahasia tersebut tidak diketahui oleh orang lain (Sejati, 2007).

Pada steganografi *modern*, arti steganografi berkembang menjadi penyembunyian informasi pada media *file* digital berupa dokumen, citra, *audio*, dan video. Tujuan dari steganografi adalah merahasiakan atau menyembunyikan keberadaan dari sebuah pesan tersembunyi atau sebuah informasi (Alatas, 2009).

2.2.1 Konsep dan Terminologi

Pada steganografi menggunakan beberapa istilah penting. Istilah-istilah tersebut adalah sebagai berikut :

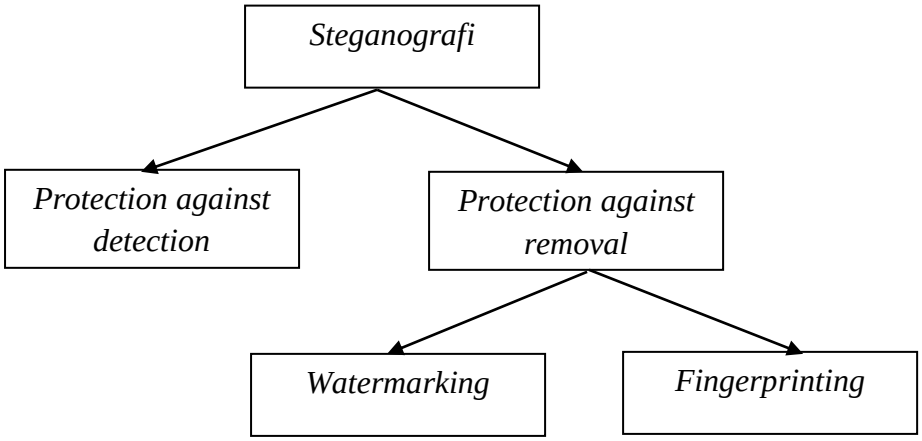
- *hiddentext* atau *embedded message* adalah pesan yang disembunyikan
- *cover image* adalah media citra yang digunakan untuk menyembunyikan *embedded message*
- *stego image* adalah pesan atau media yang sudah berisi *embedded message*.

Gambar 2.1 menampilkan konsep dasar steganografi. Penyisipan pesan ke dalam media *cover image* dinamakan *embedding*, sedangkan pemisahan pesan dari *stego image* dinamakan *extracting* (Wandani, 2012).

Gambar 2.1 Konsep dasar steganografi

2.2.2 Pembagian pada Steganografi

Pada Gambar 2.2 disajikan bahwa proteksi pada steganografi terbagi menjadi dua model atau jenis yaitu :



Gambar 2.2 Pembagian steganografi

1. *Protection against detection*

   Model proteksi ini banyak digunakan dalam dunia maya sebagai *security tools* dalam pengiriman data atau dokumen melalui internet atau media lainnya. Proteksi ini mempunyai metode agar suatu *file*/media sampul yang telah disisipi data tidak dapat dideteksi oleh steganalisis, sehingga data yang dikirimkan aman sampai orang yang ditujukan.

2. *Protection against removal*

   Model proteksi ini banyak digunakan dalam media *digital security*. Biasanya model ini berfungsi sebagai penanda hak cipta (*copyright*) agar tidak dapat dihilangkan maupun diganti oleh pihak-pihak lain yang tidak bertanggung-jawab. Pada metode ini terdapat dua metode yang digunakan, yaitu *watermarking* dan *finger printing*.

   Watermarking merupakan suatu bentuk metode dari steganografi dalam mempelajari teknik-teknik bagaimana penyimpanan suatu data *digital* kedalam data sampul *digital* yang lain. Parameter-parameter yang ada dalam penerapan metode *watermarking* adalah jumlah data yang disembunyikan (*bit rate*), ketahanan terhadap proses pengolahan sinyal (*robustness*), tak terlihat atau *output* tidak berbeda dengan *input* awal (Sejati, 2007).

2.2.3 Kriteria Steganografi

Kriteria pada penyembunyian pesan menggunakan teknik steganografi sebagai berikut :

1. *Fidelity*

   Mutu dari citra penampung tidak banyak berubah. Implementasi dari steganografi tidak menurunkan kualitas atau bahkan merusak medianya supaya tidak menimbulkan kecurigaan.

2. *Imperectibility*

   Keberadaan pesan rahasia dalam media penampung tidak dapat dideteksi oleh inderawi. Misalnya, jika media steganografi berupa citra, maka penyisipan pesan membuat citra *stegoimage* sukar dibedakan oleh mata dengan citra aslinya.

3. *Recovery*

   Data yang disembunyikan harus dapat diungkap kembali. Karena tujuan steganografi adalah data *hiding*, maka sewaktu-waktu data rahasia dalam citra penampung harus dapat diambil kembali untuk digunakan lebih lanjut (Edisuryana, 2013).

2.2.4 Metode *End Of File* (EOF)

Metode yang digunakan pada *digital watermarking* beragam tetapi secara umum metode ini menggunakan *redundant bits* sebagai tempat menyembunyikan pesan pada saat dilakukan kompresi data. Kemudian menggunakan kelemahan indera manusia yang tidak sensitif sehingga tidak ada perbedaan yang terlihat atau yang terdengar. Metode *End Of File* merupakan salah satu metode yang digunakan dalam steganografi. Metode ini menggunakan cara dengan menyisipkan data pada akhir *file* (Sejati, 2007).

Dalam metode *End Of File*, data yang disisipkan pada akhir *file* diberi tanda khusus sebagai pengenal *start* dari data tersebut dan pengenal akhir dari data tersebut. Metode *End Of File* merupakan sebuah metode yang diadaptasi dari metode penanda akhir *file* (*End Of File*) yang digunakan oleh sistem operasi windows. Dalam sistem operasi windows, jika ditemukan penanda *End Of File*

pada sebuah *file*, maka sistem akan berhenti melakukan pembacaan pada *file* tersebut. Prinsip kerja *End Of File* menggunakan karakter atau simbol khusus yang diberikan pada setiap akhir *file* (Anggraini dan Dolly, 2014).

Misalkan pada sebuah citra berukuran 5x5 *pixel* disisipkan pesan yakni "gajah". Nilai desimal ASCII dari pesan diberikan sebagai berikut :

103	**97**	**106**	**97**	**104**

Misalkan matrik nilai desimal dari *pixel* citra sebagai berikut :

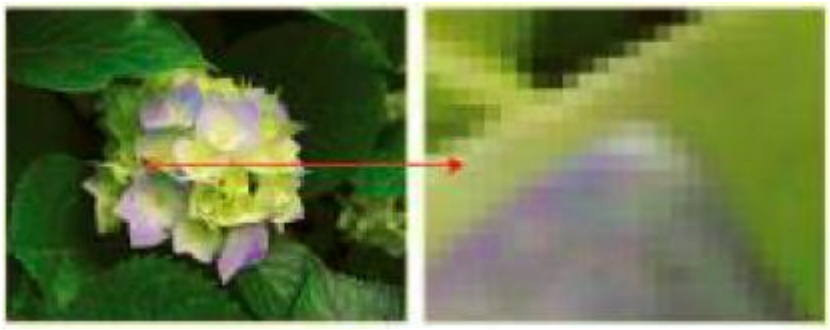
162	78	123	212	110
35	76	191	148	68
221	234	107	56	98
122	76	112	92	107
271	178	154	177	102

Nilai desimal pesan disisipkan pada akhir citra menjadikan nilai berikut :

162	78	123	212	110
35	76	191	148	68
221	234	107	56	98
122	76	112	92	107
271	178	154	177	102
103	**97**	**106**	**97**	**104**

2.3 Citra Digital

Citra digital adalah data yang ditampilkan dalam bentuk gambar sehingga memiliki arti tertentu. Sebuah citra digital menyimpan data berupa bit yang dapat dimengerti oleh manusia dengan visualisasi bit tertentu pada kanvas menjadi gambar. Pengolahan data yang dapat dilakukan terhadap citra digital antara lain adalah menampilkan bentuk gambar, melakukan perubahan pada gambar, dan mencetak citra digital pada media kertas. Citra digital terdiri dari *pixel-pixel* berukuran kecil yang membentuk sebuah gambar yang dapat dilihat oleh mata manusia. Kepadatan *pixel-pixel* yang ada dalam gambar disebut dengan resolusi. Semakin besar resolusi sebuah citra digital maka kualitas gambar tersebut semakin baik (Anggraini dan Dolly, 2014).



Gambar 2.3 *Pixel* pada citra digital

Citra digital terdiri dari tiga warna dasar, yaitu merah, hijau, dan biru (RGB). Setiap elemen pada citra digital (elemen matriks) disebut *image element* dan *picture element* atau *pixel*. Citra digital dapat didefinisikan sebagai fungsi dua variabel, f(x,y), dimana x dan y adalah koordinat spasial dan nilai f(x,y) adalah intensitas citra pada koordinat tersebut. Citra digital berukuran M x N biasanya

dinyatakan dalam bentuk matriks yang berukuran M baris dan N kolom (Astuti, 2013).

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & .. & f(0,N-1) \\ f(1,0) & f(1,1) & .. & f(1,N-1) \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ f(M-1,0) & f(M-1,1) & .. & f(M-1,N-1) \end{bmatrix}$$

Ichsan (2011) menerangkan beberapa format citra yang dapat digunakan sebagai media steganografi, yakni :

1. JPG/JPEG (*Joint Photograph (Expert) Group*)

   JPG adalah jenis data yang dikembangkan oleh *Joint Photographic Experts Assemble* (JPEG) yang banyak digunakan untuk menyimpan gambar-gambar dengan ukuran lebih kecil. Beberapa karakteristik gambar JPEG sebagai berikut.

   a. Memiliki ekstensi .jpg atau .jpeg.
   b. Mampu menayangkan warna dalam kedalaman 24-bit *true color*.
   c. Mengkompresi gambar dengan sifat *lossy*.
   d. Umumnya digunakan untuk menyimpan gambar-gambar hasil foto.

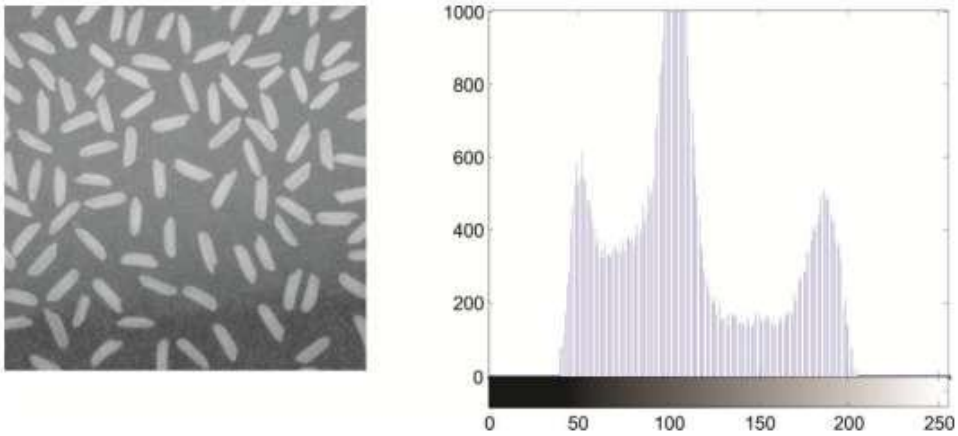
2. PNG (*Portable Network Graphics*)

   PNG adalah salah satu format citra yang menggunakan metode pemadatan yang tidak menghilangkan bagian citra tersebut (*lossless compression*). PNG dibaca “ping” namun biasanya dieja apa adanya untuk menghindari kerancuan dengan istilah “ping” pada jaringan komputer. Format PNG diperkenalkan untuk menggantikan format penyimpanan citra GIF. Secara umum format PNG digunakan sebagai citra *web*. Tipe PNG merupakan

solusi kompresi yang *powerful* dengan warna yang lebih banyak (24 bit RGB + *alpha*). Kelebihan *file* PNG adalah adanya warna transparan dan *alpha*. Warna *alpha* memungkinkan sebuah gambar transparan, tetapi gambar tersebut masih dapat dilihat mataseperti samar-samar atau bening.

2.4 *Histogram*

Histogram dalam pengolahan citra merupakan grafik yang menunjukkan distribusi dari intensitas citra. *Histogram* citra menyatakan frekuensi kemunculan berbagai derajat keabuan dalam citra. Berikut adalah contoh untuk *histogram* pada citra.



Gambar 2.4 *Histogram* citra

Histogram memiliki empat proses, yakni :

1. Apabila gambar gelap maka *histogram* cenderung ke sebelah kiri
2. Apabila gambar terang maka *histogram* cenderung ke sebelah kanan.
3. Apabila gambar *low contrast* maka *histogram* mengumpul di suatu tempat.
4. Apabila gambar *high contrast* maka *histogram* merata di semua tempat.

Informasi yang terdapat di dalam *histogram*, yakni :

1. Puncak *histogram* yaitu intensitas yang paling menonjol.
2. Lebar puncak yaitu rentang kontras.
3. Citra yang baik mengisi derajat keabuan secara penuh dan merata pada setiap nilai intensitas *pixel*.
4. *Over-exposed* (terlalu terang) dan *under exposed* (terlalu gelap) memiliki rentang kontras sempit (Anggraeni, 2014)

2.5 *System Development Life Cycle* (SDLC)

Software Development Life Cycle atau *System Development life Cycle* merupakan serangkaian aktivitas yang digunakan untuk mengatur proses pengembangan sistem. Pada SDLC terdapat dua pendekatan utama, yakni :

1. Pendekatan prediktif

   Pendekatan prediktif adalah pendekatan yang dilakukan dengan asumsi dari awal proyek sudah dapat diprediksi dengan baik.

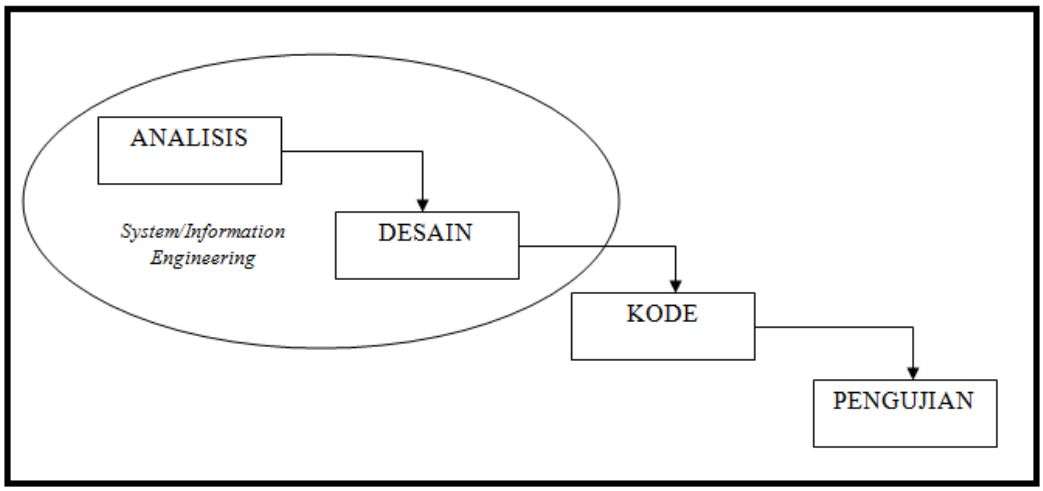
2. Pendektan adaptif

   Pendekatan adaptif adalah pendekatan yang lebih fleksibel dengan asumsi proyek tidak dapat direncanakan dari awal secara rinci.

Terdapat banyak model yang dapat digunakan dalam membangun sistem berbasis SDLC yaitu model *Waterfall, Prototype, Rapid Aplication Development (RAD), Iterative* dan *Spiral* (Satzinger et al, 2007).

2.5.1 SDLC Model *Waterfall*

Pressman (2002) menerangkan model *waterfall* merupakan metode perangkat lunak yang sistematik dan sekuensial yang mulai pada tingkat dan kemajuan sistem sampai pada analisis, desain, pengkodean, dan pengujian. Tahapan model *waterfall* disajikan pada Gambar 2.5.



Gambar 2.5 Tahapan model *Waterfall*

1. Analisis

   Proses menganalisis dan pengumpulan kebutuhan sistem yang sesuai proses bisnis sistem dan antarmuka (*interface*) yang diperlukan.

2. Desain

   Proses merancang desain dan model sistem yang akan dikembangkan berdasarkan tahap analisis pada tahap sebelumnya.

3. Kode

   Proses menerjemahkan desain ke dalam suatu bahasa yang dimengerti oleh komputer.

4. Pengujian

   Proses pengujian berfokus pada logika internal *software*, memastikan bahwa semua pernyataan sudah diuji, dan pada eksternal fungsional, yaitu mengarahkan pengujian untuk menemukan kesalahan-kesalahan dan memastikan bahwa input yang dibatasi akan memberikan hasil aktual yang sesuai dengan hasil yang dibutuhkan.

2.6 Metode Berorientasi Objek

Metode berorientasi objek adalah suatu strategi perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metode berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, Perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek (Satzinger et al, 2007).

Keuntungan menggunakan metodelogi berorientasi objek adalah sebagai berikut :

1. Meningkatkan produktifitas

   Kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reusable*) sehingga meningkatkan produktifitas.

2. Kecepatan pengembangan

   Analisis dan perancangan dilakukan dengan baik dan benar agar mengurangi kesalahan pada saat pengkodean sehingga mempercepat pengembangan perangkat lunak.

3. Kemudahan pemeliharaan

   Model objek dan pola-pola yang stabil dapat dipisahkan dengan pola-pola yang sering berubah-ubah sehingga memudahkan pemeliharaan.

4. Adanya konsistensi

   Notasi pada analisis, perancangan, dan pengkodean digunakan dan diwariskan secara konsisten pada tiap tahapnya.

5. Meningkatkan kualitas

   Perangkat lunak dikembangkan dengan pendekatan dunia nyata dan adanya konsistensi pada saat pengembangannya. Perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan sehingga kualitas perangkat lunak baik.

2.7 *Unified Modelling Languange* (UML)

UML dirilis tahun 1995 sebagai sebuah metode untuk menggambarkan desain *software*. UML merupakan metode standar bahasa pemodelan untuk dokumentasi *software* berorientasi objek. UML mengedepankan penggunaan diagram untuk menggambarkan aspek dari sistem yang sedang dimodelkan (Sugiarti, 2013).

Keuntungan dari penggunaan UML diantaranya sebagai berikut :

1. *Software* didesain dan didokumentasi secara profesional sebelum dibuat.

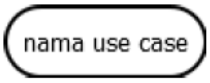
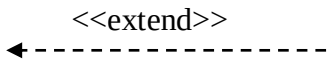
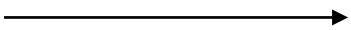
2. Karena mendesain terlebih dahulu, maka *reusable code* dapat dikode dengan tingkat efisiensi yang tinggi.
3. ‘Lubang’dapat ditemukan pada saat penggambaran desain.
4. Dapat melihat gambaran besar dari suatu *software*.

UML menjanjikan dan menghasilkan hasil dengan biaya rendah, *software* lebih efisien, lebih dapat dipercaya, dan hubungan antar bagian yang telibat menjadi lebih baik. Terdapat banyak diagram yang dapat digunakan pada UML antara lain *object diagram, class diagram, component diagram, composite structure diagram, package diagram, deployment diagram, use case diagram, activity diagram, state machine diagram, sequence diagram, communication diagram, timing diagram,* dan *interaction overview diagram.*

2.7.1 *Use Case Diagram*

Use case diagram adalah pemodelan untuk tingkah laku (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan antara satu atau lebih *actor* dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Berikut adalah simbol-simbol yang digunakan pada *use case* diagram (Sugiarti, 2013).

Tabel 2.1 Simbol-simbol *use case diagram*

Simbol	Keterangan
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau *actor*; biasanya dinyatakan dengan menggunakan kata kerja frase nama *use case*.
Actor 	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar dari sistem yang dibuat.
Asosiasi	Komunikasi antar *actor* dan *use case* yang berpartisipasi pada *use case* atau *use case* yang memiliki interaksi dengan *actor*.
Ekstensi 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walaupun tanpa *use case* tambahan itu.
Generalisasi 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari yang lain.

Menggunakan / *include* / *uses* <<include> <<uses>>>	Relasi *use case* tambahan ke sebuah *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

2.7.2 *Activity Diagram*

Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Perlu diperhatikan bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan *actor*. Berikut merupakan simbol-simbol yang terdapat pada *activity diagram* (Sugiarti, 2013).

Tabel 2.2 Simbol-simbol *activity diagram*

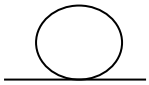
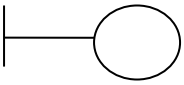
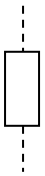
Simbol	Keterangan
Status awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.

Penggabungan/*join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

2.7.3 *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display/form*) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan. Berikut merupakan simbol-simbol yang terdapat pada *sequence diagram* (Sugiarti, 2013).

Tabel 2.3 Simbol-simbol *sequence diagram*

Simbol	Keterangan
Aktor 	Menggambarkan orang yang sedang berinteraksi dengan sistem.
Entity Class 	Menggambarkan hubungan kegiatan yang akan dilakukan.
Boundary Class 	Menggambarkan sebuah penggambaran dari *form*.
Control Class 	Menggambarkan penghubung antara *boundary* dengan tabel.
Focus of Control & Life Line 	Menggambarkan tempat mulai dan berakhirnya sebuah *message*.
Message *a message*	Menggambarkan pengiriman pesan.