BAB II

LANDASAN TEORI

2.1 *Game*

Dalam kamus bahasa Indonesia "*Game*" diartikan sebagai permainan. Permainan adalah kegiatan yang kompleks yang didalamnya terdapat peraturan, play dan budaya. Sebuah permainan adalah sebuah sistem dimana pemain terlibat dalam konflik buatan, disini pemain berinteraksi dengan sistem dan konflik dalam permainan merupakan rekayasa atau buatan, dalam permainan terdapat peraturan yang bertujuan untuk membatasi perilaku pemain dan menentukan permainan.

Video game merupakan salah satu media hiburan yang paling popular untuk semua kalangan usia. Sejak pertama kali ditemukan sampai saat sekarang, teknologi *game* telah mengalami kemajuan yang terbilang sangat pesat. Hal ini ditandai dengan berkembangnya jenis, produk, alat, dan jenis interaksi *game* dengan penggunaan yang semakin beragam bentuknya.

2.1.1 Pengertian *Game* Menurut Ahli

Menurut Agustinus Nilwan (1995:1), *game* merupakan permainan komputer yang dibuat dengan teknik dan metode animasi. Jika ingin mendalami pengunaan animasi haruslah memahami pembuatan *game*. Atau jika ingin membuat *game*, maka haruslah memahami teknik dan metode animasi, sebab keduanya saling berkaitan.

2.2 Sejarah *Game*

Pada tahun 1952, seorang mahasiswa Universitas Cambridge bernama A.S. Gouglas membuat permainan OXO (tic-tac-toe) dalam versi grafik. Permainan ini ia kembangkan ketika hendak mendemonstrasikan tesisnya tentang interaksi antara manusia dan komputer. Pada tahun 1958 William Higin Botham mendesain sebuah *game* dengan judul *Tennis For Two* yang dimainkan dalam *oscilloscope*, dan kemudian ada pula Steve Russel pada tahun 1961 dengan *game* berjudul *Spacewar* yang dibuat dalam komputer mainframe DEC PDP-1 saat mereka menjalani studi di MIT. Sejarah video *game* tidak hanya mengenai orang-orang yang berperan di dalamnya, tetapi juga mengenai berbagai perusahaan *game* yang mempunyai kasus ironis. Atari adalah perusahaan Amerika dengan nama Jepang, dan perusahaan Jepang bernama SEGA didirikan oleh orang Amerika. Magnavox yang memulai peran perusahaan ini telah berumur 1 abad, dan Nintendo sebagai perusahaan yang mempopulerkan kembali video *game* juga sama tuanya, serta tidak ada yang pernah berpikir bahwa Sony yang merupakan perusahaan penemu banyak barang-barang elektronik mulai dari transistor radio hingga alat perekam video, akan membuat sebuah *console* yang menjadi produk dengan tingkat penjualan yang tinggi hingga saat ini.

2.2.1 Era sebelum munculnya *game* (1889 1970)

Era ini merupakan era paling awal dimana merupakan masa pertama kali tumbuhnya perusahaan-perusahaan *game* yang terkenal hingga saat ini dan mulai ditemukannya bentuk video *game* yang pertama. Dimulai pada tahun 1889 dimana

Fusajiro Yamauchi mendirikan Marufuku sebagai perusahaan yang memproduksi dan mendistribusikan Hanafuda, permainan kartu ala Jepang.

Kemudian perusahaan ini mengganti nama menjadi *The Nintendo Playing Card Company*, di mana kata Nintendo berarti menyerahkan keberuntungan pada surga. Tahun 1954 seorang bernama David Rosen, melihat popularitas dari mesin permainan berbasis koin yang dilihatnya di markas Amerika di Jepang. Oleh karena itu ia mendirikan Service *Game*s untuk mengeksport permainan ini ke Jepang. Pada tahun 1960an Rosen memutuskan untuk membuat sendiri mesin permainan berbasis koin miliknya dan akhirnya membeli Tokyo Jukebox and *Slotmachine Company*. Nama SEGA yang merupakan singkatan dari kata *Service Games* akhirnya diadopsi menjadi nama perusahaannya.

Sekitar tahun 1960an banyak *game* yang dijalankan dengan komputer universitas di Amerika Serikat, yang dikembangkan oleh beberapa pengguna individu yang memprogramnya pada waktu senggang. Namun jumlah *game* yang dibuat sedikit dan banyak yang tidak mengenalnya atau memainkannya. Pada tahun 1961, Steve Russell dan kawan-kawan memprogram permainan komputer berjudul *Spacewar* di DEC. Permainan komputer ini akhirnya didistribusikan dengan komputer DEC dan diperdagangkan di *cyberspace primitif*.

Permainan komputer ini juga dipresentasikan di MIT saat *Science Open House* tahun 1962. Ralph Baer, ketika di *Sanders Associates*, membuat video *game* sederhana yang diberi nama *Chase* yang dapat ditampilkan di televisi standar. Baer, bersama Bill Harison, mengembangkan *light gun*. Dan bersama Bill Rusch, Baer membuat video *game*s pada tahun 1967. Ia mengembangkan

penemuannya dan akhirnya pada tahun 1968 ia menghasilkan *prototype* yang dapat memainkan beberapa permainan komputer termasuk di dalamnya permainan komputer tenis meja dan menembak.

2.2.2 Era mulainya *game* (1971-1977)

Pada tahun 1971 Nolan Bushnell dan Ted Dabney membuat versi coin*operated arcade* dari *Spacewar* dan mesin itu dinamainya *Space Computer*. Pada tahun yang sama Bushnell dan Dabney memutuskan untuk membuat sebuah perusahaan yang membuat video *game*, yang kemudian didistribusikan melalui perusahaan lain yang diberi nama Atari, diambil dari istilah permainan Jepang Go yang hampir sama dengan istilah *Check* pada catur.

Bushnell membuat sendiri programnya berupa video *game* tennis sebagai yang disebut Pong. Pong menjadi permainan komputer terlaris setelah Atari berdiri dengan lisensinya sendiri, dan menjadi pelopor ratusan permainan komputer tiruan. Pada tahun yang sama ketika Atari berdiri GregoryYo*b* menulis *Hunt The Wumpus*, sebuah permainan computer petak umpet yang ditengarai menjadi *text adventure game* pertama. Yob menulis itu sebagai reaksi permainan petak-umpet yang sudah beredar di pasaran seperti *Hurkle, Mugwump*, dan *Snark*. Pada tahun 1975 Will Crowther menulis *text adventure game* pertama yang kita kenal sampai sekarang. *Adventure* (judul aslinya *Advent*, dan selanjutnya disebut *Collosal Cave*) *game* tersebut diprogram di Fortran untuk PDP-10.

Pemain mengkontrol permainan dengan kalimat sederhana seperti perintah-perintah dan nantinya akan mendapat deskripsi teks sebagai output. *Death Race* (1976) yang dikembangkan oleh Exidy menjadi kontroversi karena

kekerasan yang dimasukkan dalam permainan komputer ini. Kontoversi ini meningkatkan awareness publik yang tidak akan pernah berhenti untuk diperdebatkan bahkan sampai saat ini. Pada tahun 1977 Atari mengeluarkan home console, Atari 2600, dengan menggunakan *cartridge* yang banyak dikenal sebagi media menyimpan permainan komputer sampai era tahun 2000. 9 permainan komputer didesain dan dilempar ke pasaran untuk musim liburan.

2.2.3 Masa Keemasan (1978-1981)

Masa ini merupakan masa keemasan bagi industri *arcade*, jaman keemasan ini ditandai dengan meratanya bisnis arcade dan munculnya arcade ame berwarna yang berlanjut sampai pertengahan tahun 80an dan 90an. Dimulai dengan dirilisnya *Space Invaders* oleh Taito tahun 1978.

Game ini mendongkrak banyak perusahaan untuk memasuki pasar arcade dan memproduksi *game* yang mereka buat sendiri. Juga pada tahun 1978, Atari mengeluarkan *Asteroids*. Permainan komputer tersebut adalah permainan komputer terlaris, menggeser kedudukan *Lunar Lander*, yang pernah dikeluarkan Atari. Permainan komputer arcade berwarna semakin terkenal pada tahun 1979 dan 1980. Permainan komputer klasik lain di penghujung tahun 1970 diantaranya *Night Driver, Galaxian* dan *Breakout*.

2.3 Multimedia

Konsep multimedia telah banyak diterapkan dalam dunia pendidikan. Pembelajaran menggunakan multimedia interaktif berkembang atas dasar pembelajaran konvensional yang tidak bisa memenuhi kebutuhan peserta didik dalam pembelajaran, Sigit, dkk, (2008:29). Multimedia dapat menjadikan suatu aplikasi menjadi sangat interaktif dan menyajikan interface yang menarik.

2.3.1 Pengertian Multimedia

Multimedia merupakan kombinasi dari teks, gambar, seni grafik, suara, animasi dan elemen-elemen video yang dimanipulasi secara digital. Tampilan dan cita rasa dari proyek multimedia harus menyenangkan, estetis, mengundang dan mengikat. Proyek harus memuat konsistensi visual, hanya dengan menggunakan elemen-elemen yang mendukung pesan keseluruhan dari program. Vaughan, (2006,p3).

2.3.2 Jenis Multimedia

Menurut Sigit, dkk. (2008:1) multimedia terbagi menjadi dua jenis, yaitu:

1. Multimedia Linier
   Multimedia linier adalah suatu multimedia yang tidak dilengkapi oleh alat pengontrol apapun yang dapat dioperasikan oleh pengguna. Multimedia ini berjalan sekuensial (berurutan), contohnya: TV dan film.
2. Multimedia Interaktif
   Multimedia interaktif dilengkapi oleh alat pengontrol yang dapat dioperasikan oleh pengguna, sehingga pengguna dapat memilih apa yang dikehendaki untuk proses selanjutnya. Contohnya: *video game,* pembelajaran interaktif, dan lain-lain.

2.3.3 Kelebihan Pembelajaran Menggunakan Multimedia

Menurut Townsend seperti dikutip oleh Husni Idris (2008:1), multimedia memliki beberapa keuntungan, yaitu:

1. Multimedia masuk akal sehingga dapat meningkatkan pembelajaran.
2. Multimedia meningkatkan ekpresi diri dengan membiarkan pelajar untuk memutuskan sendiri.
3. Multimedia membuat pelajar menjadi ’pemilik’ sehingga mereka bisa menciptakan apa yang hendak mereka pelajari.
4. Multimedia menciptakan suasana yang aktif, sehingga pelajar dapat terlibat langsung.
5. Multimedia dapat menjembatani komunikasi pelajar dengan instruktur.

6. Pemakaian multimedia sudah tidak asng lagi karena telah digunakan dalam kehidupan sehari-hari seperti video *game* dan televisi.

2.4 Konsep Pendukung

Di dalam bagian ini terdapat beberapa konsep pendukung untuk perancangan permainan yang berdasarkan referensi dari hasil pustaka dan internet, diantaranya: defenisi *action script, Action Script pada Frame, action script* pada *button, action script* pada *movie clip*.

2.4.1 Definisi Action Script

Untuk membangun permainan ini dibutuhkan sebuah bahasa pemrograman yang ada dalam aplikasi *Macromedia Flash*, yaitu *action script*. Menurut Liman dalam blognya ia menyatakan bahwa "Action *script* adalah bahasa pemrograman yang di pakai oleh software Flash untuk mengendalikan objek-objek ataupun movie yang terdapat dalam Flash". Adapun pengertian action script menurut sebuah sumber tertulis yang berpendapat bahwa action script adalah "suatu perintah yang diletakan pada suatu frame atau objek sehingga frame atau objek tersebut akan menjadi interaktif ". Jadi objek-objek yang ada dalam *Macromedia Flash* dapat dikendalikan dengan menggunakan *action script*.

2.4.2 Action Script pada Frame

Action script pada *frame* adalah *script* yang diletakkan di dalam *frame* yang dibedakan dengan nama identitas atau *instance name.* Adapun pengertian *frame* menurut Zainul Ahmad Fanani dan Diginnovac (2007:5), ia mengungkapkan bahwa "*Frame merupakan tempat objek-objek presentasi diletakkan*". *Instance name* adalah salah satu identitas atau nama yang terpenting dalam *flash* yang berfungsi membedakan antara objek yang satu dengan yang lainnya.

Di dalam properti sebuah *symbol*, aturan pemberian nama *instance name* antara lain adalah :

1. Nama instansi tidak boleh sama dengan nama *symbol*.
2. Tidak boleh menggunakan nama yang sama dengan suatu perintah dalam *flash*, misalnya: objek, *level*, *break* dan sebagainya.
3. Nama instansi tidak boleh menggunakan spasi, titik, dan tanda baca yang lain.
4. Tidak boleh menggunakan angka saja untuk nama akan tetapi sebuah nama instansi boleh diikuti dengan angka. Contoh yang salah : 12, 13, dan sebagainya, contoh yang benar : benda1, benda_2 dan sebagainya.
5. Tidak boleh ada nama instansi yang sama.
6. Dua atau lebih *symbol* yang sama boleh dipakai, tetapi harus memiliki nama instansi yang berbeda.
7. Untuk mempermudah proses penulisan *script*, dapat ditambahkan "_mc" untuk *symbol* bertipe movieclip dan "_btn" untuk *symbol* bertipe *button*. Contoh : "benda1_mc", "tombol1_btn" dan sebagainya.

2.4.3 Action Script pada Button

Berikut adalah aturan penggunaan *action script* pada *button*, diantaranya adalah :

1. Pastikan *button* (tombol) baik yang dibuat sendiri ataupun yang mengambilnya dari *panel library*, *components* sudah terseleksi. Cara menyeleksinya yaitu dengan *selection tool* dan klik satu kali pada *button* yang ingin diberi *action*.
2. Setelah *button* diseleksi maka akan muncul keterangan.

3. Setelah properties dari *button* yang dibuat telah muncul, maka langkah selanjutnya adalah membuka *panel action*. Terdapat beberapa cara untuk membuka panel action, pertama bisa dengan langsung menekan F9, yang kedua dengan mengklik kanan dan memilih *action*, kemudian cara yang ketiga adalah langsung membukanya pada *panel action*.
4. Yang harus diingat di sini untuk mencegah terjadinya kesalahan pada *panel action* yang telah terbuka adalah terdapat keterangan yang menyebutkan bahwa *action* merupakan milik dari *button* yang telah diseleksi, karena setiap *action* berbeda-beda antara penempatan pada *button, movieclip*, ataupun pada *frame*.
5. Lalu tuliskan *script* yang ingin ditulis atau diisikan pada *button* yang telah ada.

2.4.4 *Action Script* pada *Movie Clip*

Movie Clip digunakan untuk mengatur jalannya *animasi* dalam *stage*. Dengan *movie clip animasi* dalam Flash dapat diatur sehingga *animasi* yang dibuat tidak menimbulkan keanehan. Misalnya ketika mouse ditekan, *movie clip* A bergerak dan ketika *movie clip* B ditekan dengan keyboard maka terus bergerak dan seterusnya. Kini anda yang menjadi sutradara dalam pembuatan *animasi* ini. Anda dapat menentukan *movie clip* mana yang bergerak, mana yang diam atau beberapa *movie clip* dijalankan bersama-sama.

1. Buat *layer* Bola dan *Action*
2. Buat objek lingkaran dan F8 → *Movie clip* di *layer* Bola. Buat *animasi tween motion*

3. Tambahkan frame di layer Action-frame 40. Di layer Action, klik kanan objek lingkaran→Action, tambahkan script stop, caranya seperti di langkah sebelumnya
4. Klik Global Functions→Movie Clip Control→klik On Clip Event 2x. Setelah itu pilih mouseDown dan klik 2x play
5. Tekan Ctrl + Enter untuk menjalankan animasinya.

2.5 Algoritma

Dalam matematika dan komputasi, algoritma atau algoritme merupakan kumpulan perintah untuk menyelesaikan suatu masalah. Perintah-perintah ini dapat diterjemahkan secara bertahap dari awal hingga akhir. Masalah tersebut dapat berupa apa saja, dengan catatan untuk setiap masalah, ada kriteria kondisi awal yang harus dipenuhi sebelum menjalankan algoritma. Algoritma akan dapat selalu berakhir untuk semua kondisi awal yang memenuhi kriteria, dalam hal ini berbeda dengan heuristik. Algoritma sering mempunyai langkah pengulangan (iterasi) atau memerlukan keputusan (logika Boolean danperbandingan) sampai tugasnya selesai.

Desain dan analisis algoritma adalah suatu cabang khusus dalam ilmu komputer yang mempelajari karakteristik dan performa dari suatu algoritma dalam menyelesaikan masalah, terlepas dari implementasi algoritma tersebut. Dalam cabang disiplin ini algoritma dipelajari secara abstrak, terlepas dari sistem komputer atau bahasa pemrograman yang digunakan. Algoritma yang berbeda dapat diterapkan pada suatu masalah dengan kriteria yang sama.

Kompleksitas dari suatu algoritma merupakan ukuran seberapa banyak komputasi yang dibutuhkan algoritma tersebut untuk menyelesaikan masalah. Secara informal, algoritma yang dapat menyelesaikan suatu permasalahan dalam waktu yang singkat memiliki kompleksitas yang rendah, sementara algoritma yang membutuhkan waktu lama untuk menyelesaikan masalahnya mempunyai kompleksitas yang tinggi.

2.5.1 Sejarah Algoritma

Kata algoritma berasal dari latinisasi nama seorang ahli matematika dari Uzbekistan Al Khawārizmi (hidup sekitar abad ke-9), sebagaimana tercantum pada terjemahan karyanya dalam bahasa latin dari abad ke-12 "Algorithmi de numero Indorum".

Pada awalnya kata algoritma adalah istilah yang merujuk kepada aturan-aturan aritmetis untuk menyelesaikan persoalan dengan menggunakan bilangan numerik arab (sebenarnya dari India, seperti tertulis pada judul di atas).

Pada abad ke-18, istilah ini berkembang menjadi algoritma, yang mencakup semua prosedur atau urutan langkah yang jelas dan diperlukan untuk menyelesaikan suatu permasalahan. Masalah timbul pada saat akan menuangkan bagaimana proses yang harus dilalui dalam suatu/sebuah sistem (program) bagi komputer sehingga pada saat eksekusinya, komputer dapat bekerja seperti yang diharapkan. Programer komputer akan lebih nyaman menuangkan prosedur komputasinya atau urutan langkah proses dengan terlebih dahulu membuat gambaran (diagram alur) diatas kertas.

2.6 Jenis-jenis Algoritma

2.6.1 Algoritma BFS

BFS merupakan metode pencarian solusi dimana semua node pada level n akan dikunjungi terlebih dahulu sebelum mengunjungi node-node pada level n+1. Pencarian dimulai dari node akar terus ke level 1 dari kiri ke kanan, kemudian berpindah ke level berikutnya dari kiri ke kanan hingga solusi ditemukan.

2.6.2 Algoritma Branch And Bound

Branch and bound merupakan metode yang membagi permasalahan menjadi subregion yang mengarah ke solusi (branching) dengan membentuk sebuah struktur pohon pencarian (search tree) dan melakukan pembatasan (bounding) untuk mencapai solusi optimal. Proses branch merupakan membangun semua cabang tree yang menuju solusi, sedangkan proses bound merupakan menghitung node dengan memperhatikan batas constraint.

2.7 UML

UML adalah bahasa untuk menspesifikasi, memvisualisasi, membangun dan mendokumentasikan artifacts (bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak, artifact tersebut dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya.

2.7.1 Use Cases

Use Case merupakan sebuah teknik yang digunakan dalam pengembangan sebuah software atau sistem informasi untuk menangkap kebutuhan fungsional

dari sistem yang bersangkutan, Use Case menjelaskan interaksi yang terjadi antara ‘aktor’ – inisiator dari interaksi sistem itu sendiri dengan sistem yang ada, sebuah Use Case direpresentasikan dengan urutan langkah yang sederhana.

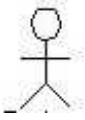
Aktor adalah sesuatu atau seseorang yang ada di luar sistem dan ikut berperan serta dalam aktifitas sistem. Aktor bisa berupa: End User, perangkat hardware, bahkan sistem yang lain. Setiap use case merupakan sebuah seri yang lengkap dari sebuah event kejadian, dilihat dari sudut pandang aktor.

Fokus dari sebuah use case adalah menjelaskan bagaimana mencapai sebuah tujuan atau goal. Dalam pengembangan software, dibutuhkan banyak use case untuk mendefinisikan scope dari sebuah system. derajat formalitas dari sebuah sistem yang dikembangkan menentukan level detail yang dibutuhkan dari sebuah use case.

Use Case sebaiknya jangan dicampuradukan dengan fitur dari sistem, sebuah use case mungkin berhubungan dengan satu atau lebih fitur sistem, sebuah fitur mungkin terelasi dengan satu atau lebih use case.

Agar kita mengerti mengenai Use Case diagram, maka pertama-tama kita harus mengerti dulu elemen-elemen yang ada di dalamnya.

1. Actor:

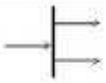
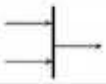
   elemen aktor diumpamakan adalah sebuah peran dari ‘user’ sebuah sistem. aktor ini bisa berupa ‘human user’ atau bahkan bisa sebuah sistem yang lain. dalam diagram use case, elemen aktor dinotasikan dengan notasi .

karakteristik dari sebuah aktor antara lain:

- Aktor merupakan elemen eksternal dari sistem
- Aktor berinteraksi dengan sistem, aktor boleh menggunakan fungsionalitas dan menerima informasi yang disediakan oleh sistem dan aktor juga dapat memberikan informasi kepada sistem.
- Aktor dapat berbentuk class, maka aktor bisa memiliki instance atau objek yang merepresentasikan aktor yang lebih spesifik.

2.7.2 Activity Diagram

Activity diagram memiliki pengertian yaitu lebih fokus kepada menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Dipakai pada business modeling untuk memperlihatkan urutan aktifitas proses bisnis. Memiliki struktur diagram yang mirip flowchart atau data flow diagram pada perancangan terstruktur. Memiliki pula manfaat yaitu apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan. Dan activity dibuat berdasarkan sebuah atau beberapa use case pada use case diagram.

Simbol	Keterangan
	Start Point
	End Point
	Activities
	Fork (Percabangan)
	Join (Penggabungan)
	Decision
Swimlane	Sebuah cara untuk mengelompokkan activity berdasarkan Actor (mengelompokkan activity dalam sebuah urutan yang sama)

Gambar 2.1 Simbol-simbol Activity Diagram

Terdapat beberapa hal penting yang harus diketahui, yaitu ;

- Activity mengambarkan sebuah pekerjaan atau tugas dalam workflow
- Pada UML, activity digambarkan dengan simbol belah ketupat



- Start state dengan tegas menunjukan dimulainya suatu workflow pada sebuah activity diagram
- Hanya ada satu start state dalam sebuah workflow
- Pada UML, start state digambarkan dengan simbol lingkaran yang solid



Start State

- End state menggambarkan akhir atau terminal dari pada sebuah activity diagram
- Bisa terdapat lebih dari satu end state pada sebuah activity diagram
- Pada UML, end state digambarkan dengan simbol sebuah bull's eye

End State

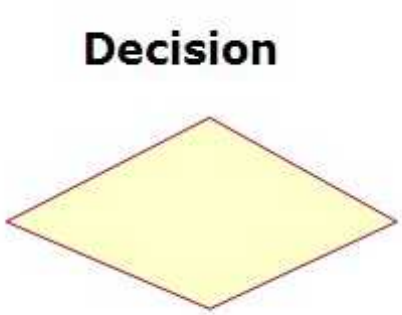


- State transition menunjukan kegiatan apa berikutnya setelah suatu kegiatan sebelumnya
- Pada UML, state transition digambarkan oleh sebuah solid line dengan panah



State Transition

- Decision adalah suatu titik atau point pada activity diagram yang mengindikasikan suatu kondisi dimana ada kemungkinan perbedaan transisi
- Pada UML, decision digambarkan dengan sebuah simbol diamond



2.7.3 Class Diagram

Class diagram menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti hubungan hubungan satu sama lain seperti hubungan dinamis, pewarisan, asosiasi, dan agregasi.

Tabel 2.1 Class Diagram

Nama	Keterangan	Simbol
Class	*Class* adalah blok - blok pembangun pada pemrograman berorientasi obyek. Sebuah *class* digambarkan sebagai sebuah kotak yang terbagi atas 3 bagian. Bagian atas adalah bagian nama	Nama *Class* + atribut + atribut + atribut
Association	Sebuah asosiasi merupakan sebuah relationship paling umum antara 2 *class* dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 *class*. Garis ini bisa melambangkan tipe-tipe	

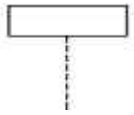
Composition	Jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka *class* tersebut memiliki relasi *Composition* terhadap *class* tempat dia bergantung tersebut.	
Dependency	Kadangkala sebuah *class* menggunakan *class* yang lain. Hal ini disebut *dependency.* Umumnya penggunaan *dependency* digunakan untuk	

2.7.4 Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa message yang digambarkan terhadap waktu. Sequence diagram terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons 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.

Tabel 2.2 Sequence Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi