BAB II

TINJAUAN PUSTAKA

II.1 Macromedia Flash

II.1.1 Pengertian Flash

Menurut Wandah wibawanto flash merupakan program grafis multimedia dan animasi yang dapat dipergunakan untuk keperluan pembuatan aplikasi web yang interaktif dan menarik. Memainkan suatu game pada komputer sangatlah menyenangkan, akan tetapi bagi sebagian orang akan lebih menyenangkan membuat sendiri suatu game dan membiarkan orang lain memainkan game buatannya (Wandah Wibawanto :2005;1).

Menurut Vaughan Tay Multimedia merupakan kombinasi teks, suara, animasi dan video yang disampaikan kepada anda dengan komputer atau peralatan manipulasi elektronik dan digital yang lain. Multimedia dapat menimbulkan sensasi yang dahsyat. (Vaughan Tay:2006).

II.2 Simulasi

II.2.1 Pengertian Simulasi

Simulasi merupakan salah satu cara untuk memecahkan berbagai persoalan yang dihadapi didunia nyata (*real world).* Banyak metode yang dibangun dalam *operation research* dan *system analyst* untuk kepentingan pengambilan keputusan dengan menggunakan berbagai analisis data.(Thomas J. Kakiay: 2004)

Metode simulasi merupakan salah satu metode mengajar yang dapat digunakan dalam pembelajaran kelompok. Proses pembelajaran yang

menggunakan simulasi cenderung objeknya bukan benda atau kegiatan yang sebenarnya, melainkan kegiatan mengajar yang bersifat pura-pura. Kegiatan simulasi dapat dilakukan oleh siswa pada kelas tinggi di Sekolah Dasar. Dalam pembelajaran, siswa akan dibina kemampuannya berkaitan dengan keterampilan berinteraksi dan berkomunikasi dalam kelompok. Disamping itu, dalam metode simulasi siswa diajak untuk bermain peran beberapa perilaku yang dianggap sesuai dengan tujuan pembelajaran.

Ada beberapa jenis model simulasi di antaranya, yaitu:

1. Bermain peran (role playing) dalam proses pembelajarannya metode ini mengutamakan pola permainan dalam bentuk dramatisasi. Dramatisasi dilakukan oleh kelompok siswa dengan mekanisme pelaksanaan yang diarahkan oleh guru untuk melaksanakan kegiatan yang telah ditentukan / direncanakan sebelumnya. Simulasi ini lebih menitik beratkan pada tujuan untuk mengingat atau menciptakan kembali gambaran masa silam yang memungkinkan terjadi pada masa yang akan datang atau peristiwa yang aktual dan bermakna bagi kehidupan sekarang.

2. Sosiodramad alam pembelajarannya yang dilakukan oleh kelompok untuk melakukan aktivitas belajar memecahkan masalah yang berhubungan dengan masalah individu sebagai makhluk sosial. Misalnya, hubungan anak dan orangtua, antara siswa dengan teman kelompoknya.

3. Permainan simulasi (Simulasi games) dalam pembelajarannya siswa bermain peran sesuai dengan peran yang ditugaskan sebagai balajar membuat suatu keputusan.

Dalam menciptakan sebuah aplikasi, terdapat beberapa hal yang perlu diperhatikan guna perolehan hasil yang maksimal(Whitten *et al*, 2005),antara lain sebagai berikut :

a. Produktivitas

Saat ini hampir segala bidang memerlukan aplikasi yang dapat di gunakan sesuai dengan keperluan dalam bidangnya.Hal ini menyebabkan permintaan terhadap pengadaan aplikasi lebih banyak. Dan tuntutan terhadap kualitas aplikasi yang lebih bagus dan handal. Ternyata hal ini membutuhkan lebih banyak programmer dan penganalisa sistem yang berkualitas, kondisi kerja extra, kemampuan pemakai untuk mengembangkan sendiri, bahasa pemograman yang lebih baik, perawatan sistem yang lebih baik, disiplin teknis perangkat lunak.

b . Reliabilitas

Relibilitas suatu perangkat tidak seperti faktor kualitas lain yang dapat di ukur, di arahkan dan dietimasi dengan menggunakan data pengembangan histrois. Relibilitas perangkat lunak di defenisikan dalam bentuk statistik sebagai kemungkinan operasi program computer bebas kegagalan di dalam suatu lingkungan dalam kurun waktu tertentu.

c. Maintabilitas

Maintabilitas mencakup perawatan aplikasi, seperti :

- Koreksi jika ditemukan kesalahan pada program.
- Pengadaptasian jika lingkungan berubah.
- Modifikasi jika pengguna membutukan perubahan kebutuhan.

d. Integritas

Integritas adalah mengukur kemampuan sistem suatu aplikasi untuk menahan serangan terhadap sekuritasnya.Dalam hal ini kekuatan sistem akan di uji terhadap serangan dari tipe tertentu yang dapat terjadi suatu waktu.

e. Usabilitas

Usabilitas merupakan ukuran terhadap kualitas interaksi yang terjadi antara aplikasi dengan pengguna.Ukuran usabilitas dapat di ketahui melalui tampilan fisik suatu aplikasi (*user friendly*), penggunaan waktu yang efesien dan lain sebagainya.

II.1.2 Karakteristik Metode Simulasi

Karakteristik sistem yang akan di rancang dalam skripsi ini adalah sistem yang terotomasi, yang merupakan bagian dari sistem buatan manusia dan berinteraksi atau di control oleh satu atau lebih computer sebagai bagian dari sistem yang di gunakan dalam masyarakat modern.

Menurut Pohan (1997), sistem terotomasi mempunyai sejumlah komponen yaitu :

a. Perangkat keras, antara lain CPU,disk,terminal,printer dan perangkat keras pendukung lainnya. Sedangkan perangkat lunaknya antara lain sistem operasi,sistem database,program aplikasi dan lain sebagainya.
b. Personil, antara lain pengguna sistem, menyediakan masukan, mengkonsumsi keluaran, dan melakukan aktivitas manual yang mendukung sistem.

c. Data,merupakan segala sesuatu yang harus tersimpan dalam sistem, selama jangka waktu tertentu, dan prosedur, antara lain intruksi dan kebijakan untuk mengoperasikan sistem.

II.1.3 Metode Simulasi

Menurut Law dan Kelton Metode Simulasi merupakan suatu teknik meniru operasi-operasi atau proses- proses yang terjadi dalam suatu sistem dengan bantuan perangkat komputer dan dilandasi oleh beberapa asumsi tertentu sehingga sistem tersebut bisa dipelajari secara ilmiah (Law and Kelton: 1991).

Menurut Bonett Satya Lelono Djati model simulasi merupakan salah satu alat dari analisis kuantitatif yang sangat populer. Keandalan simulasi mampu menghadapi kompleksitas permasalahan, mengukur kinerja dari suatu data yang bervariasi dan mampu memberikan solusi alternatif secara cepat lewat bantuan program komputer (Bonett Satya Lelono Djati: 2005)

II.1.4 Pengertian Multimedia

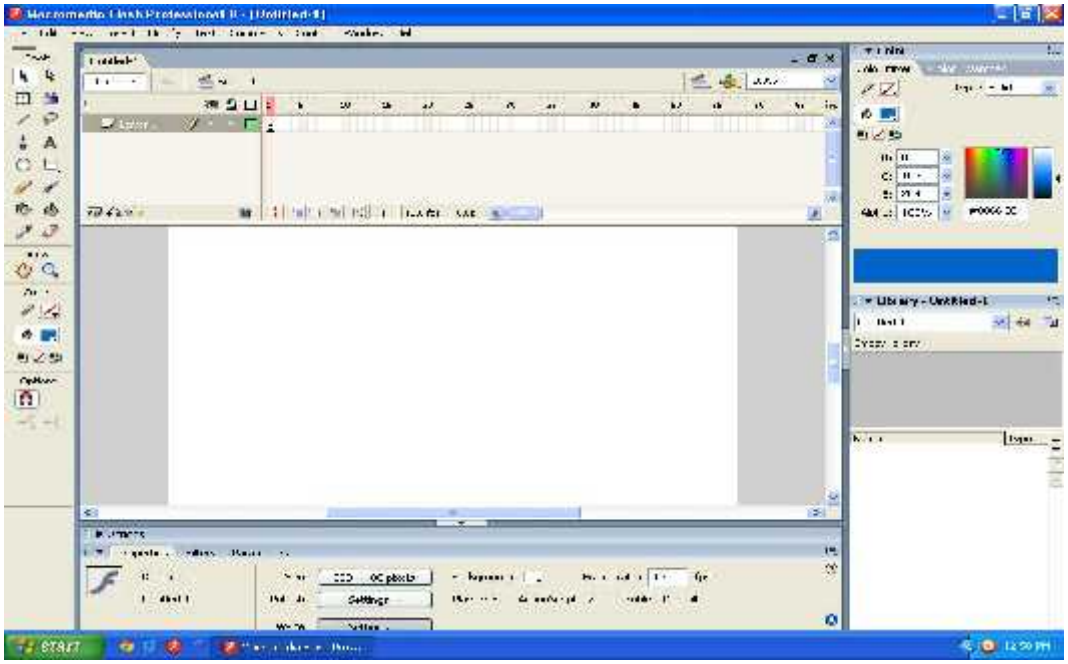
Menurut Rachmat dan Alphone Secara etimologis multimedia berasal dari kata multi (Bahasa Latin) yang berarti banyak, bermacam-macam dan *medium* (Bahasa Latin) yang berarti sesuatu yang dipakai untuk menyampaikan atau membawa sesuatu. Kata *medium* dalam *American Heritage Electronic Dictionary* (1991) juga diartikan sebagai alat untuk mendistribusikan dan mempresentasikan informasi (Rachmat dan Alphone: 2006).

II.2 Macromedia Flash 8

Macromedia Flash adalah sebuah program animasi yang telah banyak digunakan para animator untuk menghasilkan animasi yang *professional*. Diantara program-program animasi yang ada, *macromedia flash* merupakan program yang paling fleksibel dalam pembuatan animasi seperti animasi interaktif, game, *company*, profil, presentasi, *movie* dan tampilan animasi lainnya.(MADCOMS: 2007).

II.2.1 Mengenal *Macromedia Flash 8*

Macromedia Flash merupakan sebuah program yang didesain khusus oleh *Macromedia* saat itu**.** Sebagai pengembangannya yang saat ini telah dibeli oleh *Adobe Incorporated* sehingga berubah nama menjadi *Adobe Flash. Flash* didesain dengan kemampuan untuk membuat animasi 2D (dua dimensi) yang handal dan ringan sehingga *flash* banyak digunakan untuk membangun dan memberikan efek animasi pada website, CD interaktif dan yang lainnya. Tampilan *Macromedia Flash 8* dapat dilihat pada gambar II.1.



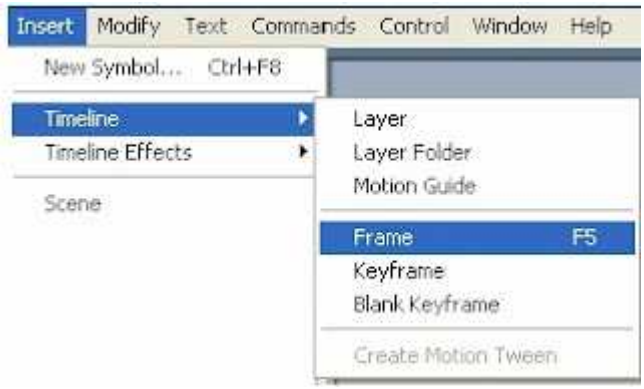
Gambar II.1 ***Macromedia Flash 8***

Sumber: Mahir Dalam 7 Hari Macromedia Flash Pro 8, MADCOMS, 2007

Bagian-bagian penting dalam area kerja di atas diantaranya: Menu, Toolbox, Timeline, Stage dan Panel.

1. *Menu*

Menu pada Macromedia Flash Pro 8 terdiri dari: File, Edit, View, Insert, Modify, Text Commands, Control, Window dan Help. Anda dapat melihat submenu yang terdapat pada masing-masing menu dengan mengeklik satu kali pada menu yang ingin Anda pilih.



Gambar II.2 *Menu*

Sumber: Mahir Dalam 7 Hari Macromedia Flash Pro 8, MADCOMS, 2007

2. *Toolbox*

Dalam toolbox terdapat komponen-komponen penting diantaranya: Tools, View, Colors dan Options. Toolbox memiliki peran untuk memanipulasi atau memodifikasi objek dalam stage.Berikut adalah tampilan dari Tool Box.

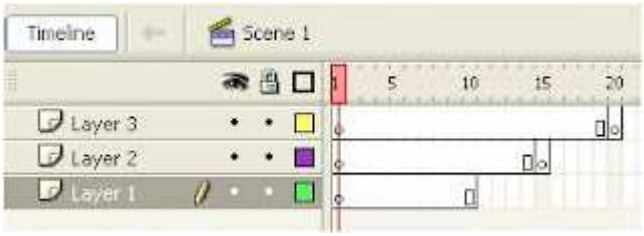
Gambar II.3 *Tool Box*

Sumber: Mahir Dalam 7 Hari Macromedia Flash Pro 8, MADCOMS, 2007

3. *Time Line*

Timeline atau garis waktu merupakan komponen yang digunakan untuk mengatur atau mengontrol jalannya animasi. Timeline terdiri dari beberapa layer. Layer digunakan untuk menempatkan satu atau beberapa objek dalam stage agar dapat

diolah dengan objek lain. Setiap layer terdiri dari frame-frame yang digunakan untuk mengatur kecepatan animasi. Semakin panjang frame dalam layer, maka semakin lama animasi akan berjalan.



Gambar II.4 *Time Line*

Sumber: Mahir Dalam 7 Hari Macromedia Flash Pro 8, MADCOMS, 2007

5. Panel

Beberapa panel penting dalam *Macromedia Flash Pro 8* diantaranya panel *Properties, Filters, Parameters, Actions, Library, Color, Align, Info* dan *Transform.*

a. Panel *Properties, Filters, Parameters*

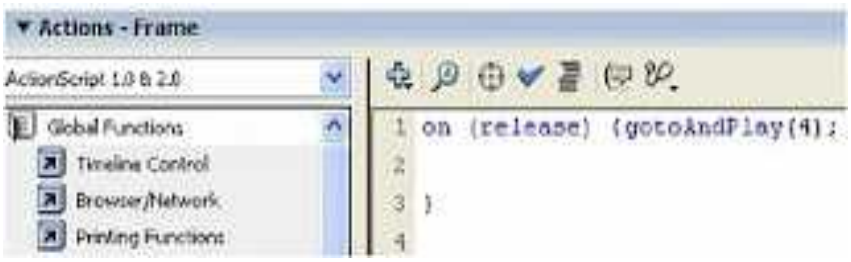
Panel ini terdapat di bawah stage. Untuk mengeluarkan atau menyembunyikan panel ini dapat digunakan shortcut Ctrl+F3. Panel Properties & Filters & Parameters digunakan untuk untuk mengatur ukuran background, warna background, kecepatan animasi dan lain-lain.

Gambar II.5 Panel *Properties, Filters, Parameters*

Sumber: Mahir Dalam 7 Hari Macromedia Flash Pro 8, MADCOMS, 2007

b. Panel *Actions*

Panel Actions digunakan untuk menuliskan script atau bahasa pemrograman flash (ActionScript). Anda dapat mengetikkan secara langsung pada layar Actions atau menggunakan bantuan yang disediakan oleh Macromedia Flash Pro 8. Untuk memunculkan atau menyembunyikan panel ini dapat digunakan shortcut F9.



Gambar II.6 Panel *Actions Script*

Sumber: Mahir Dalam 7 Hari Macromedia Flash Pro 8, MADCOMS, 2007

c. Panel *Library*

Library merupakan panel yang digunakan untuk menyimpan objek-objek berupa graphic atau gambar, button atau tombol, movie dan suara baik yang dibuat langsung pada stage ataupun hasil proses impor dari luar stage. Untuk memunculkan atau menyembunyikan panel ini dapat digunakan shortcut Ctrl+L.

Gambar II.7 Panel *Library*

Sumber: Mahir Dalam 7 Hari Macromedia Flash Pro 8, MADCOMS, 2007

d. Panel Color

Panel Color merupakan panel yang digunakan untuk memilih warna yang digunakan dalam pembuatan objek-objek pada stage. Ada dua jenis subpanel, yaitu: Color Mixer dan Swatches. Shortcut untuk Color Mixer adalah Shift+F9 dan shortcut untuk Color Swatches adalah Ctrl+F9.

Gambar II.8 Panel *Color*

Sumber: Mahir Dalam 7 Hari Macromedia Flash Pro 8, MADCOMS, 2007

e. Panel *Align, Info, Transform*

Untuk menampilkan panel ini Anda dapat menekan Ctrl+K pada keyboard. Panel ini digunakanuntuk mengatur posisi objek, ingin diletakkan pada tengah

stage, sebelah kiri atau kanan dan lain-lain.Dengan panel ini Anda juga dapat memutar objek dengan Transform.

Gambar II.9 Panel *Align, Info, Transform*

Sumber: Mahir Dalam 7 Hari Macromedia Flash Pro 8, MADCOMS, 2007

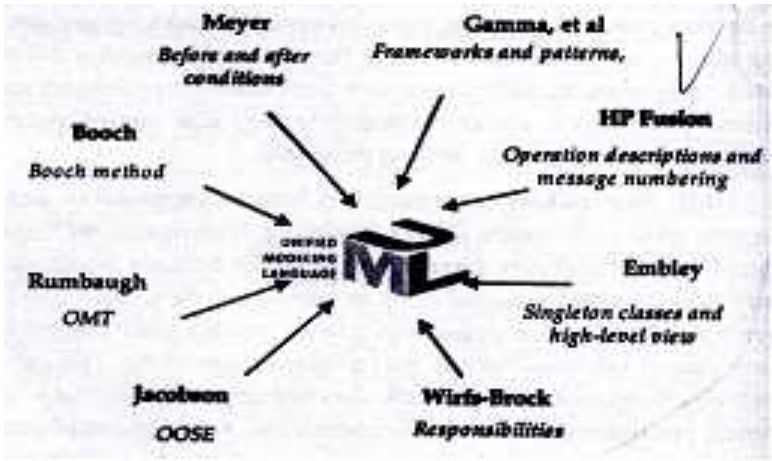
II.3 UML (*Unified Modelling Language*)

UML (*Unified Modelling Language*) adalah salah satu alat bantu yang sangat handal di dunia perkembangan sistem yang berorientasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi perkembangan sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (sharing) dan mengkomunikasikan rancangan dengan baik (Munawar ; 2005 : 17).

UML merupakan kesatuan bahasa pemodelan yang dikembangkan oleh Booch, *Object Modeling Technique* (OMT) dan *Object Oriented Engineering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented*. Metode ini menjadikan proses analisis dan design ke dalam empat tahapan interatif, yaitu: identifikasi kelas-kelas dan objek-objek,

identifikasi semantik dari hubungan objek dan kelas tersebut, perincian *interface* dan implementasi.Keunggulan metode Booch adalah pada detail dan kayanya dengan notasi dan elemen. Pemodelan OMT yang dikembangkan oleh Rumbaugh didasarkan pada analisis terstuktur dan pemodelan *entity-relationship*. Tahapan utama dalam metodologi ini adalah analisis, disain sistem, desain objek dan implementasi. Keunggulan metode ini adalah dalam penotasian yang mendukung semua konsep OO. Metode OOSE dari Jacobson lebih memberi penekanan dan *use case*. OOSE memiliki tiga tahapan yaitu membuat model *requirement* dan analisis, desain dan implementasi dan model pengujian (test Model). Keunggulan metode ini adalah mudah dipelajari karena memiliki notaasi sederhana namun mencakup seluruh tahapan dalam rekayasa perangkat lunak.

Dengan UML, metode Booch, OMT dan OOSE digabungkan dengan elemen-elemen dari metode lain yang lebih efektif dan elemen-elemen baru yang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam dari pada metode lainnya. Unsur-unsur yang membentuk UML ditunjukkan dalam Gambar II.2

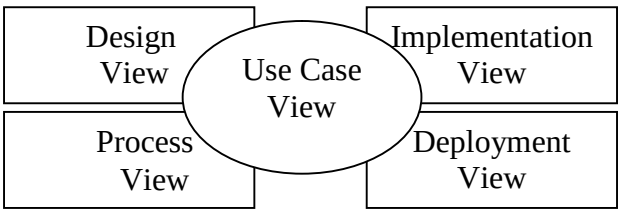


Gambar II.10 Unsur-unsur yang membentuk UML

Sumber : Pemodelan Visual dengan UML, Munawar, 2005 : 18

UML adalah hasil kerja dari konsorsium berbagai organisasi yang berhasil dijadikan sebagai standar baku dalam OOAD (*Object Oriented Analysis* dan *Design*). UML tidak hanya domain dalam penotasian dilingkungan OO tetapi juga populer di luar lingkungan OO. Ada tiga karakter penting yang melekat di UML yaitu sketsa, cetak biru dan bahasa *pemrograman*. Sebagai sebuah sketsa UML bisa berfungsi sebagai sebuah cetak biru karena sangat lengkap dan detil. Dengan cetak biru ini maka akan bisa diketahui informasi detil tentang coding program (*Forward engineering*) atau bahkan membaca program dan mengimplementasikannya kembali ke dalam diagram (*reverse engineering*). *Reverse engineering* sangat berguna pada situasi dimana kode program yang tidak terdokkumentasi asli hilang atau bahkan belum perna dibuat sama sekali. Sebagai bahasa pemrograman, UML dapat diterjemahkan diagram yang ada di UML menjadi kode program siap untuk dijalankan.

UML dibangun atas model 4+1 *view*. Model ini didasarkan pada fakta bahhwa struktur sebuah sistem dideskripsikan dalam *view* dimana salah satu diantaranya *use case view*. *use case view* ini memegang peran khusus untuk mengintegrasikan *content* ke *view* yang lain. Model 4+1 *view* ditunjukkan pada gambar II.3



Gambar II.11 Model 4+1 *View*

Sumber : Pemodelan Visual dengan UML, Munawar, 2005 : 20

Kelima *view* tersebut tidak berhubungan dengan diagram yang dideskripsikan di UML. Setiap *view* berhubungan dengan perspektif tertentu dimana sistem akan diuji. *View* yang berbeda akan menekankan pada aspek yang berbeda dari sistem yang mewakili tentang sistem bisa dibentuk dengan menggabungkan informasi-informasi yang ada pada kelima *view* tersebut.

Use case view mendefinisikan perilaku eksternal sistem. Hal ini menjadi daya tarik bagi *end user*, analis dan tester. Pandangan ini mendefenisikan kebutuhan sistem karena mengandung semua *view* yang lain yang mendeskripsikan aspek-aspek tertentu dari peran dan sering dikatakan yang mendrive proses perkembangan perangkat lunak.

Design view mendeskripsikan struktur logika yang mendukung fungsi-fungsi yang dibutuhkan di *use case*. *Design view* ini berisi definisi komponen program, class-class utama bersama-sama dengan spesifikasi data, perilaku dan interaksinya. Informasi yang terkandung di *view* menjadi pergantian para progremer karena menjelaskan secara detil bagaimana fungsionalitas sistem akan diimplementasikan.

Implementasi *view* menjelaskan komponen-komponen visi yang akan dibangun. Hal ini berbeda dengan komponen logic yang dideskripsikan pada *design view*. Termasuk disini diantaranya *file exe*, *library* dan *database*. Informasi yang ada di *view* dan integrasi siistem.

Proses *view* berhubungan dengan hal-hal yang berkaitan dengan *concurrency do* dalam sistem. Sedangkan *deployment view* menjelaskan bagaimana komponen-komponen fisik didistribusikan ke lingkungan fisik seperti jaringan komputer dimana sistem akan dijalankan. Kedua view ini menunjukan

kebutuhan non fungsional dari sistem seperti toleransi kesalahan dan hal-hal yang berhubungan dengan kinerja (Munawar;2005:17-21).

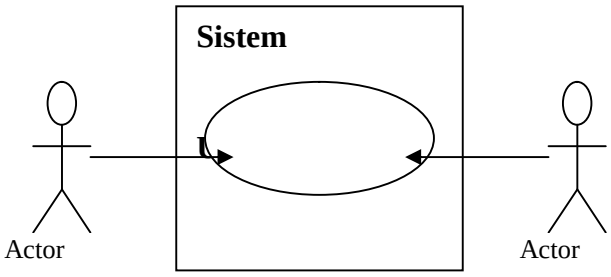
II.3.1 Use Case Diagram

Use case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara deskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem yang disebut *scenario*. Setiap *scenario* mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, sistem yang lain, perangkat keras dan urutan waktu. Dengan demikian secara singkat bisa dikatakan *use case* adalah serangkaian *scenario* yang digabungkan bersama-sama oleh pengguna tujuan umum pengguna.

Dalam pembicaraan tentang *use case*, pengguna biasanya disebut dengan *actor*. *Actor* adalah sebuah peran yang bisa dimainkan oleh pengguna dalam interaksinya dengan sistem.

Model *use case* adalah bagai dari model *requirement*. Termasuk disini adalah problem domain object dan penjelasan tentang *user interface*. *Use case* memberikan spesifikasi fungsi-fungsi yang ditawarkan oleh sistem dari *persfectif user*.

Notasi *use case* menunjukkan 3 aspek dari sistem yaitu *actor use case* dan *system / sub system boundary*. *Actor* mewakili peran orang, *system* yang lain atau alat ketika berkomunikasi dengan *use case*. Ilustrasi *actor*, *use case* dan *system* ditunjukkan pada gambar II.12



Gambar II.12 *Use Case* Diagram

Sumber : Pemodelan Visual dengan UML, Munawar, 2005 : 64

Untuk mengidentifikasi *actor*, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks targer sistem. *Actor* adalah *abstraction* dari orang dan sistem yang lain mengaktifkan fungsi dari target sistem. Orang atau sistem bila muncul dalam beberapa peran. Perlu dicatat bahwa *actor* berinteraksi dengan use case, tetapi tidak memiliki kontrol atas use *case*.

Use case adalah abstraksi dari interaksi antara sistem dan *actor*. Oleh kerena itu sangat penting untuk memilih abstraksi yang cocok. Use case dibuat berdasarkan keperluan actor. Use case harus merupakan ’apa’ yang dikerjakan software aplikasi, bukan ’bagaimana’ software aplikasi mengerjakannya. Setiap use case harus diberi nama yang menyatakan apa hal yang dicapai dari hasil interaksinya dengan actor. Namun use case boleh terdiri dari beberapa kata dan tidak boleh ada dua use case yang memiliki nama yang sama (Munawar ; 2005 : 63-66).

II.3.2 Class Diagram

Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem.

Kelas memiliki apa yang disebut *Atribut* dan metode atau operasi :

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas
2. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas

Susunan kelas suatu sistem yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

1. Kelas main, kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.
2. Kelas yang menangani tampilan sistem, kelas yang mendefinisikan dan mengatur tampilan ke pemakai.
3. Kelas yang diambil dari pendefinisian *use case*, kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case*.
4. Kelas yang diambil dari pendefinisian data, kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

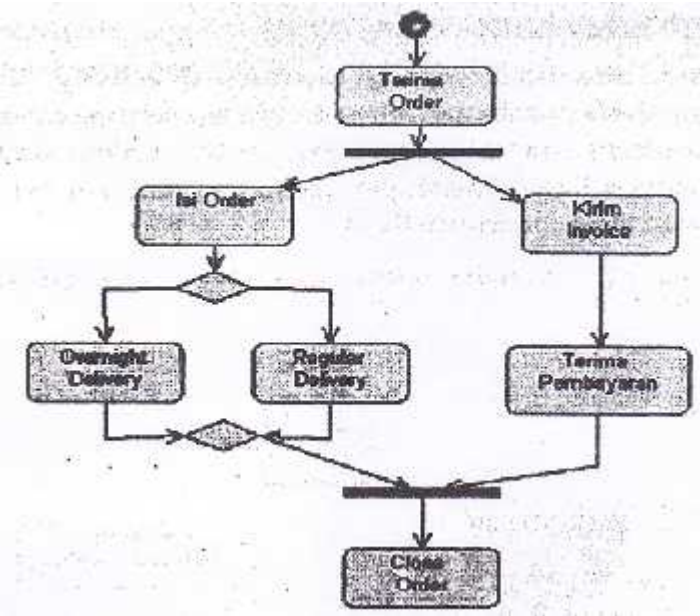
Jenis-jenis kelas diatas juga dapat digabungkan satu sama lain sesuai dengan pertimbangan yang dianggap baik asalkan fungsi-fungsi yang sebaiknya ada pada struktur kelas tetap ada. Susunan kelas juga dapat ditambahkan kelas utilitas seperti koneksi ke basis data, membaca *file* teks, dan lain sebagainya sesuai kebutuhan.

Dalam mendefinisikan metode yang ada di dalam kelas perlu memperhatikan apa yang disebut dengan *cohension* dan *coupling*. *Cohension* adalah ukuran seberapa dekat keterkaitan instruksi di dalam sebuah metode terkait satu sama lain sedangkan *coupling* adalah ukuran seberapa dekat keterkaitan instruksi antara metode yang satu dengan yang lain dalam sebuah kelas. Sebagai

aturan secara umum maka sebuah metode yang dibuat harus memiliki kadar *cohesion* yang kuat dan kadar *coupling* yang lemah (Rosa A.S dan M. Shalahuddin ; 2011 : 122-123).

II.3.3 Activity Diagram

Activity diagram adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaanya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku paralel sedangkan flowchart tidak bisa(Munawar ; 2005 : 87). Contoh *activity diagram* sederhana ditunjukkan pada gambar II.13



Gambar II.13 Contoh *Activity Diagram* Sederhana

Sumber : Pemodelan Visual dengan UML, Munawar, 2005 : 111

II.3.4 Squence Diagram

Squence diagram digunakan untuk menggambarkan perilaku pada sebuah sekenario. Diagram ini menunjukkan sebuah contoh objek dan pesan yang diletakkan diantara objek-objek ini didalam *use case*.

Komponen utama *Squence diagram* terdiri dari atas objek yang dituliskan dengan kotak segiempat bernama. *Messege* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress vertical* (Munawar ; 2005 : 109).

1. Objek / *participant*

Objek diletakkan di dekat bagian atas diagram dengan urutan dari kiri ke kanan. Mereka diatur dalam urutan guna menyederhanakan diagram. Setiap *participant* dihubungkan garis titik-titik yang disebut *lifeline.* Sepanjang *lifeline* ada kotak yang disebut *activation*. *Activation* mewakili sebuah eksekusi operasi dari *participant*. Panjang kotak ini berbanding lurus dengan durasi *activation*. *Activation* mewakili sebuah eksekusi operasi dari participant. Panjang kotak ini berbanding lurus dengan durasi *activation.* Bentuk *participant* dapat dilihat pada gambar II.14

Gambar II.14 Bentuk *Participant*

Sumber : Pemodelan Visual dengan UML, Munawar, 2005 : 88

2. *Messege*

Sebuah *messege* bergerak dari suatu *participant* ke *participant* yang lain dan dari *lifeline* ke *lifeline* yang lain. Sebuah *participant* bisa mengirim sebuah *message* kepada dirinya sendiri.

Sebuah *message* bisa jadi *simple*, *synchronous* atau *asynchoronous*. *Message* yang *simple* adalah sebuah perpindahan (transfer), contoh dari satu *participant* ke *participant* yang lainnya. Jika suatu *participant* mengirimkan sebuah *message* tersebut akan ditunggu sebelum di proses dengan urusannya. Namun jika *message asynchoronous* yang dikirimkan, maka jawabannya atas *message* tersebut tidak perlu ditunggu. Simbol *message* pada *squnence diagram* dapat dilihat pada gambar II.15

Gambar II.15 Bentuk *Messege*

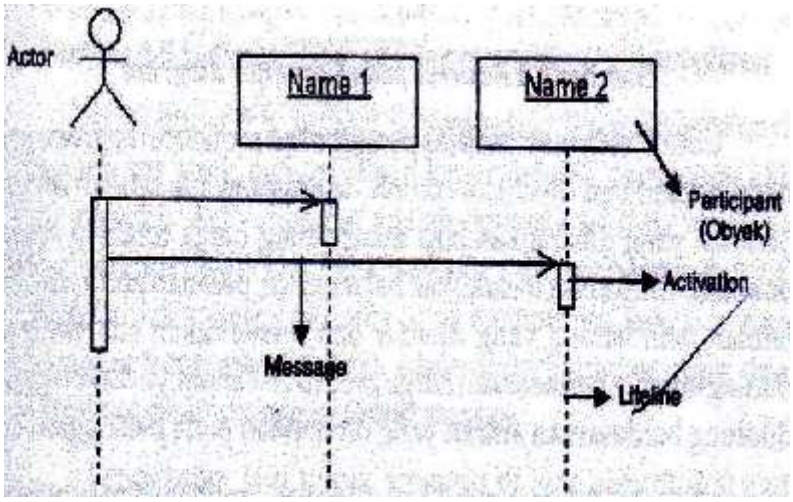
Sumber : Pemodelan Visual dengan UML, Munawar, 2005 : 88

3. *Time*

Time adalah diagram yang mewakili waktu pada arah vertikal. Waktu dimulai dari atas ke bawah. *Message* yang lebih dekat dari atas akan dijalankan terlebih dahulu dibanding *message* yang lebih dekat kebawah.

Terdapat dua dimensi pada *squence diagram* yaitu dimensi dari kiri ke kanan menunjukkan tata letak participant dan dimensi dari atas ke bawah

menunjukkan lintasan waktu. Simbol-simbol yang ada pada *squence diagram* ditunjukkan pada gambar II.16



Gambar II.16 Bentuk *Time*

Sumber : Pemodelan Visual dengan UML, Munawar, 2005 : 89