BAB II

LANDASAN TEORI

2.1 Sekilas Tentang Switch

Switch pada dasarnya mempunyai fungsi seperti Hub yaitu sebagai pembagi sinyal dan penguat sinyal pada jaringan komputer akan tetapi switch lebih cerdas dari pada Hub karena Switch dapat mengenali alamat data yang harus ditransmisikan dan mampu mengatur lalu lintas data dalam jaringan secara lebih baik dibandingkan dengan Hub.

Switch merupakan titik percabangan dari proses transfer data sehingga jika switch mengalami masalah maka seluruh koneksi jaringan dan proses transfer data akan terganggu. Switch biasanya memiliki banyak port yang akan menghubungkan ke jaringan komputer dan port - port tersebut akan berhubungan dengan konektor RJ 45.

Gambar 2.1 Contoh *Switch*

Setiap komputer yang tergabung dalam sebuah jaringan akan ditandai dengan alamat yang berbeda-beda. Kalau diibaratkan, IP address ini seperti alamat

rumah pada sebuah jalan. Jaringan yang akan kita bangun diistalahkan sebagai jalanan, dan komputer-komputer yang tergabung dalam jaringan tersebut diibaratkan rumah-rumah. Jadi antara komputer memiliki IP address yang berbeda-beda antara satu dengan yang lainnya. Bayangkan apa jadinya bila dalam sebuah jalan terdapat dua rumah yang memiliki nomor alamat yang sama. Jika terdapat surat yang hendak dikirim ke nomor alamat yang sama tersebut, pastilah surat tersebut tidak akan pernah sampai pada tujuannya, karena bingung mana alamat yang benar. Begitu pula dengan penggunaan IP address pada setiap komputer yang ada dalam jaringan. Jika terdapat alamat yang sama.

2.2 Prinsip Pengenalan Citra

Citra dapat berbentuk foto hitam putih atau berwarna, sinyal-sinyal video seperti gambar pada monitor televisi, atau bersifat digital yang dapat langsung disimpan pada suatu pita magnetik. Menurut presisi yang digunakan untuk menyatakan titik-titik koordinat pada ranah waktu atau bidang dan untuk menyatakan nilai keabuan atau warna suatu citra, maka secara teoritis citra dapat dikelompokkan menjadi empat kelas citra, yaitu cara kontinu-kontinu, kontinu-diskret, diskret-kontinu, dan diskret-diskret; dengan label pertama menyatakan presisi dari titik-titik koordinat pada bidang citra sedangkan label kedua menyatakan presisi nilai keabuan atau warna. Kontinu dinyatakan dengan presisi takhingga, sedangkan diskret dinyatakan dengan presisi angka berhingga.

Pengubahan citra yang bersifat kontinu menjadi citra yang bersifat diskret memerlukan pembuatan kisi-kisi arah vertikal dan horisontal, sehingga diperoleh citra dalam bentuk larik dua dimensi. Proses tersebut dikenal sebagai proses digitisasi atau pencuplikan (sampling). Setiap elemen larik tersebut dikenal sebagai elemen gambar atau piksel.

2.2.1 Pengolahan Citra Digital

Pengolahan Citra merupakan proses pengolahan dan analisis citra yang banyak melibatkan persepsi visual. Proses ini mempunyai ciri data masukan dan informasi keluaran yang berbentuk citra. Proses pengolahan citra dalam bentuk digital secara umum mempertimbangkan masalah peningkatan mutu citra atau perbaikan citra.

Istilah pengolahan citra digital secara umum didefinisikan sebagai pemrosesan citra dua dimensi dengan komputer.Dalam definisi yang lebih luas, pengolahan citra digital juga mencakup semua data dua dimensi.Citra digital adalah barisan bilangan nyata maupun kompleks yang diwakili oleh bit-bit tertentu. (Abdul Kadir & Adhi Susanto, 2013)

2.2.2 Metode Deteksi Tepi

Deteksi tepi merupakan salah satu proses prapengolahan yang sering dibutuhkan pada analisis citra yang bertujuan untuk meningkatkan penampakan garis pada citra. Jadi prosesnya mempunyai sifat diferensiasi atau memperkuat komponen frekuensi tinggi. Tepi mencirikan batas - batas objek dan karena itu tepi berguna untuk proses segmentasi dan identifikasi objek di dalam citra. Tujuan operasi pendeteksi tepi adalah untuk meningkatkan penampakan garis batas suatu daerah atau objek di dalam citra . Deteksi tepi (Edge Detection) pada suatu citra adalah suatu proses yang menghasilkan tepi-tepi dari obyek-obyek citra, tujuannya adalah :

1. Untuk menandai bagian yang menjadi detail citra
2. Untuk memperbaiki detail dari citra yang kabur, yang terjadi karena error atau adanya efek dari proses akuisisi citra

Suatu titik (x,y) dikatakan sebagai tepi (edge) dari suatu citra bila titik tersebut mempunyai perbedaan yang tinggi dengan tetangganya.

Cara kerja dari deteksi tepi adalah memisahkan objek yang akan di analisa dengan menentukan sumbu x dan y pada setiap objek yang akan dideteksi. Untuk menentukan nilai x dan y objek dilihat dari jarak terhadap pojok kiri atas citra. *Range* yang tepat dengan nilai sumbu x dan y setiap objek yang akan di deteksi dan diinisialisasi dalam nilai warna pixel putih sebagai nilai yang valid dan sesuai dengan nilai objek yang akan dideteksi.

Metode deteksi tepi terdiri atas deteksi tepi dengan nilai ambang, deteksi tepi dengan gradien pertama, deteksi tepi dengan gradien kedua, deteksi tepi dengan gradien arah, deteksi tepi dengan teknik geser dan selisih citra, dan deteksi segmen-segmen garis.

2.2.3 Algoritma Deteksi Tepi

"Algoritma adalah urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis dan logis". Kata logis merupakan kata kunci dalam algoritma. Langkah-langkah dalam algoritma harus logis dan harus dapat ditentukan bernilai salah atau benar. Dalam beberapa konteks, algoritma adalah spesifikasi urutan langkah untuk melakukan pekerjaan tertentu. Pertimbangan dalam pemilihan algoritma adalah, pertama, algoritma haruslah benar. Artinya algoritma akan memberikan keluaran yang dikehendaki dari sejumlah masukan yang diberikan. Tidak peduli sebagus apapun algoritma, kalau memberikan keluaran yang salah, pastilah algoritma tersebut bukanlah algoritma yang baik. Pertimbangan kedua yang harus diperhatikan adalah kita harus mengetahui seberapa baik hasil yang dicapai oleh algoritma tersebut. Hal ini penting terutama pada algoritma untuk menyelesaikan masalah yang memerlukan aproksimasi hasil (hasil yang hanya berupa pendekatan). Algoritma yang baik harus mampu memberikan hasil yang sedekat mungkin dengan nilai yang sebenarnya. Ketiga adalah efisiensi algoritma. Efisiensi algoritma dapat ditinjau dari 2 hal yaitu efisiensi waktu dan memori. Meskipun algoritma memberikan keluaran yang benar (paling mendekati), tetapi jika kita harus menunggu berjam-jam untuk

mendapatkan keluarannya, algoritma tersebut biasanya tidak akan dipakai, setiap orang menginginkan keluaran yang cepat. Begitu juga dengan memori, semakin besar memori yang terpakai maka semakin buruklah algoritma tersebut. Dalam kenyataannya, setiap orang bisa membuat algoritma yang berbeda untuk menyelesaikan suatu permasalahan, walaupun terjadi perbedaan dalam menyusun algoritma, tentunya kita mengharapkan keluaran yang sama. Jika terjadi demikian, carilah algoritma yang paling efisien dan cepat.

Tepi (edge) adalah perubahan nilai intensitas derajat keabuan yang cepat atau tiba-tiba (besar) dalam jarak yang singkat. Tujuan mendeteksi tepi sendiri adalah untuk mengelompokkan objek-objek dalam citra, dan juga digunakan untuk menganalisis citra lebih lanjut. Ada banyak algoritma yang digunakan untuk mendeteksi tepi, salah satu diantaranya adalah deteksi tepi.

Adapun kategori algoritma deteksi tepi menurut John F. Canny adalah sebagai berikut :

1. Deteksi : Kemungkinan mendeteksi titik tepi yang benar harus dimaksimalkan sementara kemungkinan salah mendeteksi titik tepi harus diminimalkan. Hal ini diamksudkan untuk memaksimalkan rasio signal-to-noise.
2. Lokalisasi : Tepi terdeteksi harus sedekat mungkin dengan tepi yang nyata.
3. Jumlah tanggapan : Satu tepi nyata tidak harus menghasilkan lebih dari satu ujung yang terdeteksi. Dengan rumusan John F. Canny tentang

kriteria ini, maka deteksi tepi optimal untuk Kelas tepian tertentu (dikenal sebagai *step edge*).

Tabel 2.1 Algoritma Deteksi Tepi

```
function ok( x, y: integer ): boolean;
begin
result := ( x >= 0 ) and ( y >= 0 ) and ( x < b.Width ) and ( y < b.Height );
end;
begin
result := false;
if not edge[y][x] then exit;
edge[y][x] := false;
result := mag[y][x] >= thrlow;
for j := -1 to 1 do
for i := -1 to 1 do
if ok( x + i, y + j ) then
result := trace( x + i, y + j ) or result;
//jika hasil rekursi menghasilkan true
//(terhubung dengan piksel yang nilainya lebih besar dari hi-threshold)
if result then begin
pr := br.ScanLine[y];
pr[x] := rgb_hitam;//citra yang baru berwarna latar putih
end;
end;
begin
bg := gaussian( b, 1 );
result := citra_create( b.Width, b.Height );//buat citra(bitmap) baru
setlength( p, b.Height );
for j := 0 to high( p ) do
p[j] := bg.ScanLine[j];
setlength( gx, b.Height );
for j := 0 to high( gx ) do
setlength( gx[j], b.Width );
setlength( gy, b.Height );
for j := 0 to high( gx ) do
setlength( gy[j], b.Width );
setlength( mag, b.Height );
for j := 0 to high( mag ) do
setlength( mag[j], b.Width );
setlength( edge, b.Height );
for j := 0 to high( edge ) do
setlength( edge[j], b.Width );
//hitung gradien gx dan gy menggunakan operator sobel
```

```
for j := 1 to b.Height - 2 do begin
for i := 1 to b.Width - 2 do begin
gy[j][i] := ( p[j + 1][i - 1].r - p[j - 1][i - 1].r ) + 2 * ( p[j + 1][i].r - p[j - 1][i].r ) + ( p[j + 1][i + 1].r - p[j
- 1][i + 1].r );
gx[j][i] := ( p[j - 1][i + 1].r - p[j - 1][i - 1].r )
+ 2 * ( p[j][i + 1].r - p[j][i - 1].r )
+ ( p[j + 1][i + 1].r - p[j + 1][i - 1].r );
mag[j][i] := sqrt( gx[j][i] * gx[j][i] + gy[j][i] * gy[j][i] );
end;
end;
for j := 1 to b.Height - 2 do begin
for i := 1 to b.Width - 2 do begin
dir := arctan2( gy[j][i], gx[j][i] );
dx := round( cos( dir ) );
dy := round( sin( dir ) );
edge[j][i] := ( mag[j][i] >= mag[j + dy][i + dx] ) and ( mag[j][i] >= mag[j - dy][i - dx] );
if edge[j][i] then begin
edge[j + dy][i + dx] := false;
edge[j - dy][i - dx] := false;
end;
end;
end;
thrlow := 10;
thrhigh := 90;
br := result;
for j := 1 to b.Height - 2 do begin
for i := 1 to b.Width - 2 do begin
if edge[j][i] and (mag[y][x] >= thrhigh) then trace( i, j );//trace melakukan edge-linking
end;
end;
```

2.2.4 Segmentasi warna normalisasi RGB.

Segmentasi warna, ada bermacam-macam model warna. Model RGB (Red Green Blue) merupakan model yang banyak digunakan, salah satunya adalah monitor. Pada model ini untuk merepresentasikan gambar menggunakan 3 buah komponen warna tersebut. Selain model RGB terdapat juga model normalisasi

RGB dimana model ini terdapat 3 komponen yaitu, r, g, b yang merepresentasikan prosentase dari sebuah piksel pada citra digital hingga dapat mendapatkan nilai RGB pada setiap gambar yang akan dianalisa.

Langkah awal untuk membuat segmentasi warna dengan normalisasi RGB ini adalah mengurai data RGB sebanding dengan ukuran pikselnya. (Abdul Kadir & Adhi Susanto, 2013)

2.2.5 Grayscale

Grayscale atau abu-abu pada sebuah image digital adalah image yang pada setiap pixelnya hanya berisikan informasi intensitas warna putih dan hitam.

Image Grayscale memiliki banyak variasi nuansa abu-abu sehingga berbeda dengan image hitam-putih. Grayscale juga disebut monokromatik karna tidak memiliki warna lain selain variasi intensitas putih dan hitam. Sebuah image yang dijadikan Grayscale akan terkesan berbeda bila dibandingkan dengan image berwarna.

Bila kita memproses pengubahan warna image menjadi grayscale pada sebuah *Canvas,* maka proses yang bisa kita lakukan adalah dengan membaca piksel demi piksel (mengambil warna piksel), mengambil ketiga warna dasar *Red, Green, dan Blue*, kemudian membuat rata-rata dari ketiga warna tersebut, menyimpannya kembali untuk ketiga warna dasar tersebut, dan yang terakhir adalah meletakkan kembali warna piksel yang sudah dalam bentuk *grayscale* tersebut ke *Canvas.*

2.2.6 *Brightness* dan *Contrast*

Metode Sobel merupakan pengembangan metode robert dengan menggunakan filter HPF yang diberi satu angka nol penyangga. Metode ini mengambil prinsip dari fungsi laplacian dan gaussian yang dikenal sebagai fungsi untuk membangkitkan HPF. Kelebihan dari metode sobel ini adalah kemampuan untuk mengurangi noise sebelum melakukan perhitungan deteksi tepi. (Abdul Kadir & Adhi Susanto, 2013). Noise yang dimaksud disini adalah titik – titik yang bukan bagian dari citra yang merusak tampilan citra tersebut. Metode Sobel menyarankan untuk menghilangkan noise menggunakan tambahan *brightness* dan *contrast*.

Brightness atau kecerahan merupakan sifat khas persepsi visual di mana sebuah warna sumber tampak memancarkan atau memantulkan cahaya. Dengan kata lain, kecerahan adalah persepsi yang ditimbulkan oleh pencahayaan dari target visual. Ini merupakan atribut subyektif dari properti dari sebuah objek yang diamati.

Contrast adalah perbedaan pencahayaan dan / atau warna yang membuat obyek (atau perwakilannya dalam gambar atau layar) dapat dibedakan. Dalam persepsi visual dari dunia nyata, kontras ditentukan oleh perbedaan warna dan kecerahan obyek dan objek lain dalam bidang yang sama pandang. Karena sistem visual manusia lebih sensitif terhadap kontras dari pencahayaan mutlak, kita dapat melihat dunia sama terlepas dari perubahan besar dalam pencahayaan sepanjang

hari atau dari tempat ke tempat. Kontras maksimum dari suatu gambar adalah rasio kontras atau jangkauan dinamis.

2.2.7 Pixel

Pixel (picture element) adalah sebuah titik yang merupakan elemen paling kecil pada citra satelit. Angka numerik (1 byte) dari pixel disebut digital number (DN). DN bisa ditampilkan dalam warna kelabu, berkisar antara putih dan hitam (gray scale), tergantung level energi yang terdeteksi. Pixel yang disusun dalam order yang benar akan membentuk sebuah citra. Kebanyakan citra satelit yang belum diproses disimpan dalam bentuk gray scale, yang merupakan skala warna dari hitam ke putih dengan derajat keabuan yang bervariasi. Untuk PJ, skala yang dipakai adalah 256 shade gray scale, dimana nilai 0 menggambarkan hitam, nilai 255 putih. Dua gambar di bawah ini menunjukkan derajat keabuan dan hubungan antara DN dan derajat keabuan yang menyusun sebuah citra.

2.3 Aplikasi dan Tool Untuk Membuat Aplikasi

Aplikasi berasal dari kata *application* yang artinya penerapan, lamaran, penggunaan. Secara istilah aplikasi adalah program siap pakai yang dibuat untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju. Adapun beberapa pengertian aplikasi lain diantaranya :

a. Menurut Hendrayudi

Aplikasi adalah kumpulan perintah program yang dibuat untuk melakukan pekerjaan-pekerjaan tertentu. (Eko Nugroho, 2008)

b. Menurut Hengky W.Pramana

Aplikasi adalah suatu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas seperti system perniagaan, game palayanan masyarakat, periklanan, atau semua proses yang hamper dilakukan manusia. (Eko Nugroho, 2008)

c. Menurut Ibisa

Aplikasi daalah alat bantu untuk mempermudah dan mempercepat proses pekerjaan dan bukan merupakan beban bagi penggunanya. (Eko Nugroho, 2008)

2.3.1 Software Yang Digunakan Dalam Membuat Aplikasi

2.3.1.1 HTML

HTML adalah singkatan dari *hypertext markup language*. Bahasa ini merupakan bahasa pemrograman web yang menjadi dasar dari sebuah halaman website yang dibuat. Bahasa ini terdiri dari tag dan aturan-aturan yang memungkinkan dalam membuat dokumen hypertext sehingga semua tampilan

web sebenarnya merupakan hasil interpretasi semua tag HTML yang digunakan. (M. Shalahudin dan Rosa A. S., 2008)

2.3.1.2 CSS

Cascading Style Sheet (CSS) merupakan salah satu bahasa pemrograman *web* untuk mengendalikan beberapa komponen dalam sebuah *web* sehingga akan lebih terstruktur dan seragam. Sama halnya styles dalam aplikasi pengolahan kata seperti Microsoft Word yang dapat mengatur beberapa style, misalnya heading, subbab, bodytext, footer, images, dan style lainnya untuk dapat digunakan bersama-sama dalam beberapa berkas (file). Pada umumnya CSS dipakai untuk memformat tampilan halaman *web* yang dibuat dengan bahasa HTML dan XHTML.

CSS dapat mengendalikan ukuran gambar, warna bagian tubuh pada teks, warna tabel, ukuran border, warna border, warna *hyperlink*, warna *mouse over*, spasi antar paragraf, spasi antar teks, margin kiri, kanan, atas, bawah, dan parameter lainnya. CSS adalah bahasa *style sheet* yang digunakan untuk mengatur tampilan dokumen. Dengan adanya CSS memungkinkan kita untuk menampilkan halaman yang sama dengan format yang berbeda.

Ada dua sifat CSS yaitu internal dan eksternal. Jika internal yang dipilih, maka skrip itu dimasukkan secara langsung ke halaman *website* yang akan

didesain. Kalau halaman *web* yang lain akan didesain dengan model yang sama, maka skrip CSS itu harus dimasukkan lagi ke dalam halaman *web* yang lain itu.

Sifat yang kedua adalah eksternal di mana skrip CSS dipisahkan dan diletakkan dalam berkas khusus. Nanti, cukup gunakan semacam tautan menuju berkas CSS itu jika halaman *web* yang didesain akan dibuat seperti model yang ada di skrip tersebut. (Eko Priyo Utomo, 2013)

2.3.1.3 PHP

PHP adalah teknologi yang diperkenalkan tahun 1994 oleh Rasmus Lerdorf.Beberapa versi awal yang tidak dipublikasikan digunakan pada situs pribadinya untuk mencatat siapa saja yang mengakses daftar riwayat hidup onlinennya. Versi pertama digunakan oleh pihak lain pada awal tahun 1995 dan dikenal sebagai Personal Home Page Tools. Terkandung didalamnya sebuah parser engine (mesin pengurai) yang sangat disederhanakan, yang hanya mampu mengolah macro khusus dan beberapa utilitas yang sering digunakan pada pembuatan home page,seperti buku tamu, pencacah, dan hal semacamnya. Parser tersebut ditulis ulang pada pertengahan 1995 dan dinamakan PHP/FI Versiaon 2.FI(Form Interprenter) sendiri berasal dari kode lain yang ditulis juga oleh Rasmus, yang menterjemahkan HTML dari data. Ia menggabungkan script Personal Home Page Tools dengan Form Interprenter dan menambahkan dukungan terhadap server database yang menggunakan format mSQL sehingga

lahirlah PHP/FI. PHP/FI tumbuh dengan pesat, dan orang-orang mulai menyiapkan kode-kode programnya supaya bisa didukung oleh PHP.(M. Shalahudin dan Rosa A. S., 2008)

2.3.1.4 Javascript

JavaScript merupakan bahasa scripting utama di internet dan bekerja di semua browser.Berikut ini beberapa sifat dari javascript:

- Menambahkan interaktivitas ke hamalam HTML.
- Merupakan bahasa pemrograman scripting.
- Bahasa Scripting merupakan bahasa yang ringan.
- Javascript merupakan bahasa terinterpretasi.

(Betha Sidik,2011)

2.3.1.5 Jquery

Jquery merupakan salah satu pustaka yang dikembangkan dengan mengembangkan JavaScript.Kehadirannya adalah untuk memudahkan penulisan kode JavaScript.Dengan menggunakan Jquery, penulisan kode JavaScript menjadi lebih sederhana (lebih ringkas).Selain itu, yang lebih penting lagi, pembuatan web

yang interaktif dan menarik menjadi jauh lebih mudah diimplentasikan daripada kalau anda menggunakan JavaScript sendiri.(Eko Priyo Utomo, 2013)

2.3.2 Text Editor

Ada beragam fasilitas untuk menulis kode program, salah satu diantaranya adalah Text Editor. Dalam proses pembuatan aplikasi untuk mendeteksi kerusakan switch dari jarak jauh ini, saya menggunakan sistem operasi open source (Machintos) yang di dalamnya sudah tersedia fasilitas Text Editor. Jika pada sistem operasi windows, kita kenal dengan nama Note pad atau Word Pad. Kelebihan dari fasilitas ini adalah dapat menuliskan banyak sekali format penulisan kode program. Untuk menjalankan kode program aplikasi untuk mendeteksi kerusakan switch dari jarak jauh ini, harus menggunakan Browser. Dengan demikian akan terlihat hasil dari kode-kode program yang kita tulis. (Dalam hal ini saya menggunakan browser mozzila firefox, karena kompatibel dengan fasilitas Jquery).

2.3.3 MYSQL

MYSQL adalah database. Database sendiri merupakan suatu jalan untuk dapat menyimpan berbagai informasi dengan membaginya berdasarkan kategori-

kategori tertentu. Dimana informasi-informasi tersebut saling berkaitan satu dengan yang lainnya.

MYSQL bersifat RDBMS (Relational Database Management System), yang memungkinkan seorang admin dapat menyimpan banyak informasi ke dalam tabel-tabel, dimana tabel-tabel tersebut saling berkaitan satu sama lain. Keuntungan RDBMS sendiri adalah kita dapat memecah database kedalam tabel-tabel tang berbeda. Setiap tabel memiliki informasi yang saling berkaitan dengan tabel yang lainnya.

Sama dengan PHP, MYSQL bersifat opensource, yang artinya semua orang dapat menggunakannya dengan gratis. MYSQL juga bersifat *Cross Platform*, yang artinya dapat digunakan under Windows, under Machintos ataupun under Linux. (Adi Nugroho,2005)

2.4 Pemodelan dengan UML

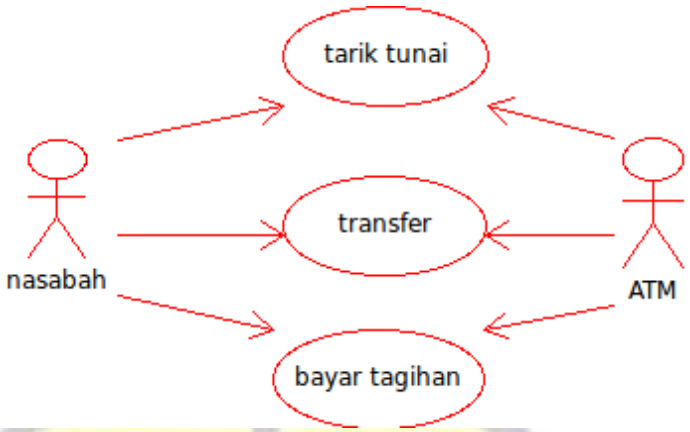
UML singkatan dari Unified Modelling Language yang berarti bahasa pemodelan standar.UML memiliki sintakas dan semantik.Ketika kita membuat model menggunakan konsep UML ada aturan-aturan yang harus diikuti. Bagaimana elemen pada model-model yang kita buat berhubungan satu dengan yang lainnya harus mengikuti standar yang ada. UML bukan hanya sekedar diagram, tetapi juga menceritakan kontekasnya. (Prabowo Pudjo Widodo, 2011)

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk :

1. Merancang perangkat lunak.
2. Sarana komunikasi antara peangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasikan system yang ada, proses-proses dan organisasinya.

2.4.1 *Use case*

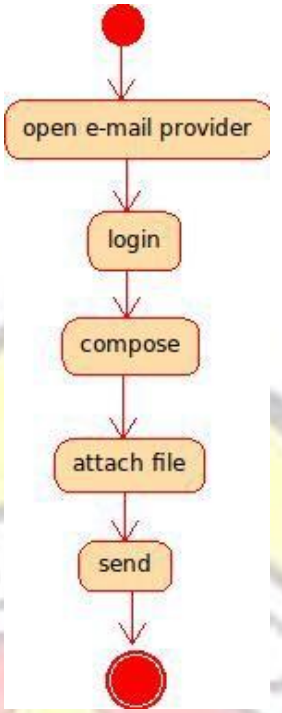
Use case diagram digunakan untuk memodelkan bisnis proses berdasarkan perspektif pengguna sistem. *Use case* diagram terdiri atas diagram untuk *use case* dan *actor*. *Actor* merepresentasikan orang yang akan mengoperasikan atau orang yang berinteraksi dengan sistem aplikasi. *Use case* merepresentasikan operasi-operasi yang dilakukan oleh *actor*. *Use case* digambarkan berbentuk elips dengan nama operasi dituliskan di dalamnya. *Actor* yang melakukan operasi dihubungkan dengan garis lurus ke *use case*.



Gambar 2.2 Contoh *Use case* (Prabowo Pudjo Widodo, 2011)

2.4.2 *Activity* Diagram

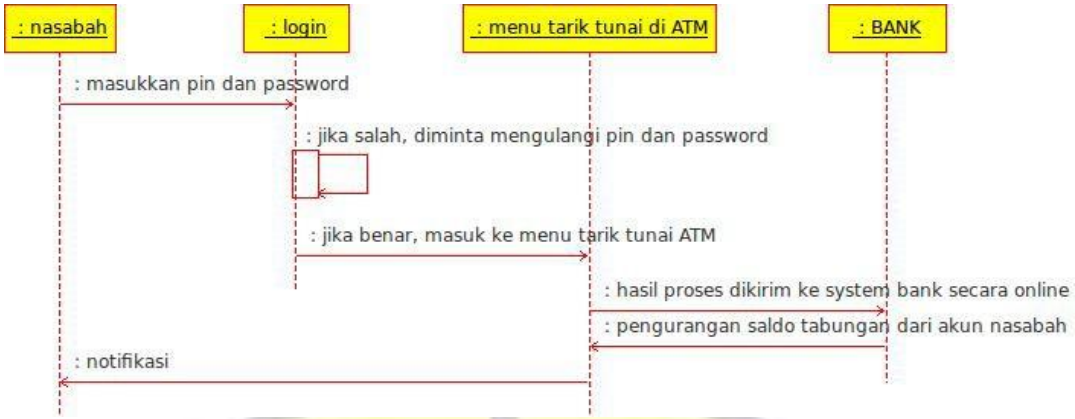
Activity diagram adalah representasi grafis dari alur kerja tahapan aktifitas. Diagram ini mendukung pilihan tindakan, iterasi dan concurrency. Pada pemodelan UML, *activity* diagram dapat digunakan untuk menjelaskan bisnis dan alur kerja operasional secara *step-by-step* dari komponen suatu sistem. *Activity* diagram menunjukkan keseluruhan dari aliran kontrol. Berikut contoh sederhana *activity* diagram pada gambar 2.3.



Gambar 2.3 Contoh *Activity* Diagram (Prabowo Pudjo Widodo, 2011)

2.4.3 *Sequence* Diagram

Sequence diagram menjelaskan secara detail urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari *use case* : interaksi yang terjadi antar class, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi Berikut contoh sederhana *Sequence* diagram pada gambar 2.4.



Gambar 2.4 Contoh *Sequence* Diagram (Prabowo Pudjo Widodo, 2011)

2.4.4. Deployment Diagram

Deployment diagram merupakan gambaran proses – proses berbeda pada suatu sistem yang berjalan dan bagaimana relasi di dalamnya. Hal inilah yang mempermudah user dalam pemakaian sistem yang telah dibuat dan diagram tersebut merupakan diagram yang statis. Misalnya untuk mendeskripsikan sebuah situs web, deployment diagram menunjukkan komponen perangkat keras (”node”) apa yang digunakan (misalnya, web server, server aplikasi dan database server), komponen perangkat lunak (”artefak”) apa yang berjalan pada setiap node (misalnya, aplikasi web, databse), dan bagaimana bagian – bagian yang berbeda terhubung (misalnya JDBC, REST, RMI).

Node digambarkan sebagai kotak, artefak yang dialokasikan ke setiap node digambarkan sebagai persegi panjang di dalam kotak. Node mungkin memiliki subnodes, yang digambarkan sebagai kotak nested. Sebuah node tunggal

secara konseptual dapat mewakili banyak node fisik seperti sekelompok database server. (Adi Nugroho,2005)