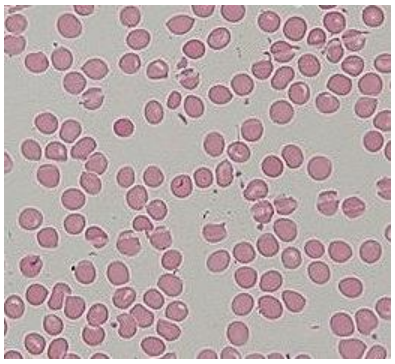
BAB II
LANDASAN TEORI

2.1 Darah

Darah adalah suatu cairan tubuh yang terdapat di dalam pembuluh darah yang warnanya merah. Darah berfungsi sebagai alat pengangkut yaitu mengambil oksigen dari paru-paru untuk diedarkan ke seluruh jaringan tubuh, mengangkut karbondioksida dari jaringan untuk dikeluarkan melalui paru-paru, mengambil zat makanan dari usus halus untuk diedarkan dan dibagikan ke seluruh jaringan tubuh, mengeluarkan zat-zat yang tidak berguna bagi tubuh untuk dikeluarkan melalui kulit dan ginjal, sebagai pertahanan tubuh terhadap serangan penyakit, menyebarkan panas ke seluruh tubuh [7].

Berdasarkan komposisinya, darah terdiri dari sekitar 99% sel darah merah (*eritrosit*), 0.6 - 1.0% keping-keping darah (*trombosit*) dan 0.2% sel darah putih (*leukosit*). Sel darah merah berfungsi untuk mengedarkan oksigen dan berperan dalam menentukan golongan darah, keping darah bertanggung jawab dalam proses pembekuan darah, sel darah putih berfungsi sebagai sistem imun tubuh dan bertugas untuk memusnahkan virus atau bakteri. Berikut ini gambar citra darah dari *Department of Histology*, Jagiellonian University Medical College:



Gambar 2. 1 Citra Darah

Perubahan komposisi pada sel-sel darah dapat menjadi petunjuk adanya kelainan atau penyakit darah. Beberapa jenis penyakit darah yang terjadi akibat

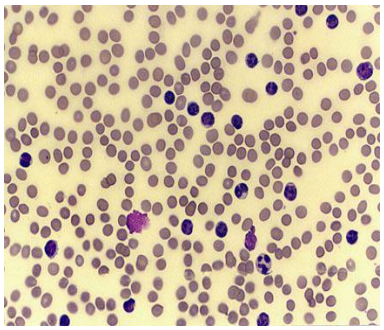
adanya perubahan komposisi pada sel-sel darah yaitu *anemia*, *leukopenia* dan *leukemia*.

2.2 Leukemia

Leukemia atau kanker darah adalah penyakit dalam klasifikasi kanker pada darah atau sumsum tulang yang ditandai oleh perubahan komposisi secara tak normal atau transformasi *maligna* dari sel-sel pembentuk darah di sumsum tulang dan jaringan limfoid, umumnya terjadi pada sel darah putih [8]. Pada tahun 2000, terdapat sekitar 256.000 orang di seluruh dunia menderita penyakit leukemia, dan 209.000 orang diantaranya meninggal karena penyakit tersebut. Sekitar 90% dari semua penderita yang terdiagnosa adalah orang dewasa.

Penyakit kanker leukemia diklasifikasikan menjadi:

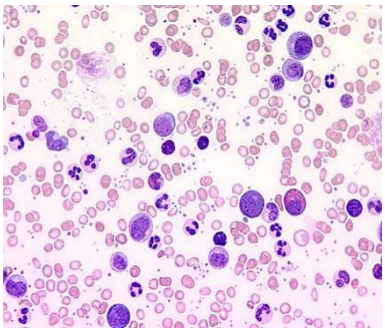
1. *Chronic Lymphocytic Leukemia* (CLL) adalah gangguan *monoklonal* ditandai dengan akumulasi progresif limfosit fungsional tidak kompeten. Pasien dengan CLL memiliki jumlah sel darah putih lebih tinggi dari biasanya. Penyakit ini sering terjadi pada orang dewasa yang berumur lebih dari 55 tahun, kadang-kadang juga diderita oleh dewasa muda, dan hampir tidak pernah terjadi pada anak-anak. Beberapa pasien meninggal dengan cepat, dalam waktu 2-3 tahun setelah didiagnosis, karena komplikasi dari CLL, tetapi kebanyakan pasien dapat bertahan hidup 5-10 tahun.



Gambar 2. 2 Chronic Lymphocytic Leukemia

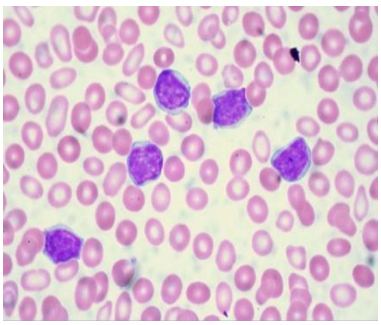
2. *Chronic Myeloid Leukemia* (CML) adalah salah satu bentuk dari *leukemia* yang ditandai dengan meningkatnya dan pertumbuhan yang tidak teratur

dari sel *myeloid* di dalam sumsum tulang dan terakumulasi juga di dalam darah. Penyakit ini sering terjadi pada orang dewasa, dapat juga terjadi pada anak-anak.



Gambar 2. 3 Chronic Myeloid Leukemia

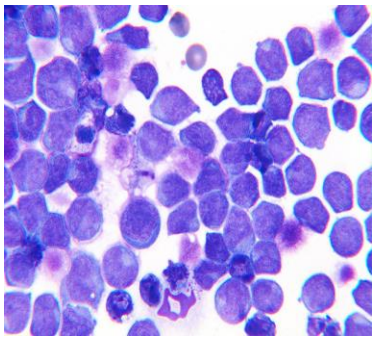
3. *Acute Lymphoblastic Leukemia* (ALL) adalah suatu penyakit dimana sel-sel yang dalam keadaan normal berkembang menjadi limfosit berubah menjadi ganas dan dengan segera akan menggantikan sel-sel normal didalam sumsum tulang. ALL merupakan leukemia yang biasa terjadi pada anak-anak dibawah umur 15 tahun. Paling sering terjadi pada anak usia antara 3-5 tahun, tetapi kadang terjadi pada usia remaja dan orang dewasa yang telah berumur 65 tahun atau lebih.



Gambar 2. 4 Acute Lymphoblastic Leukemia

4. *Acute Myelogenous Leukemia* (AML) adalah salah satu jenis kanker darah dan sumsum tulang. AML ditandai dengan pesatnya pertumbuhan sel darah putih abnormal yang menumpuk di sumsum tulang dan mengganggu

produksi sel darah normal. Penyakit ini mempengaruhi sel-sel darah yang belum matang dan berkembang dengan cepat. Penyakit ini biasa terjadi pada anak-anak dan orang dewasa.



Gambar 2. 5 Acute Myelogenous Leukemia

2.3 Kecerdasan Buatan

Kecerdasan buatan (*Artificial inteligence*) adalah salah satu cabang ilmu pengetahuan yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara yang lebih manusiawi. Hal ini biasanya dilakukan dengan mengikuti karakteristik dan analogi berpikir dari kecerdasan manusia, dan menerapkannya sebagai algoritma yang dikenal oleh komputer.

Semakin pesatnya perkembangan teknologi menyebabkan adanya perkembangan dan perluasan lingkup yang membutuhkan kehadiran kecerdasan buatan. Karakteristik cerdas sudah mulai dibutuhkan di berbagai disiplin ilmu dan teknologi. Kecerdasan Buatan tidak hanya merambah di berbagai disiplin ilmu yang lain. Irisan antara psikologi dan kecerdasan buatan melahirkan sebuah area yang dikenal dengan nama *cognition* & *psycolinguistics*. Irisan antara teknik elektro dengan kecerdasan buatan melahirkan berbagai ilmu seperti pengolahan citra, teori kendali, pengenalan pola dan robotika.

Kecerdasan buatan digunakan untuk menganalisis pemandangan dalam citra dengan perhitungan simbol-simbol yang mewakili isi pemandangan tersebut setelah citra diolah untuk memperoleh ciri khas. Kecerdasan buatan bisa dilihat sebagai tiga kesatuan yang terpadu yaitu persepsi, pengertian dan aksi. Persepsi menerjemahkan sinyal dari dunia nyata dalam citra menjadi simbol-simbol yang

lebih sederhana, pengertian memanipulasi simbol-simbol tersebut untuk memudahkan penggalian suatu informasi tertentu, dan aksi menerjemahkan simbol-simbol yang telah dimanipulasi menjadi sinyal lain yang dapat merupakan hasil akhir [9].

2.4 Pengolahan Citra (*Image Processing*)

Untuk melakukan pengenalan suatu objek berdasarkan tekstur yang dimiliki objek tersebut maka dibutuhkanlah suatu teknik untuk pengolahan citra. Pengolahan citra adalah pemrosesan citra dengan menggunakan komputer untuk menghasilkan citra yang kualitasnya lebih baik. Umumnya, operasi-operasi pada pengolahan citra diterapkan pada citra bila :

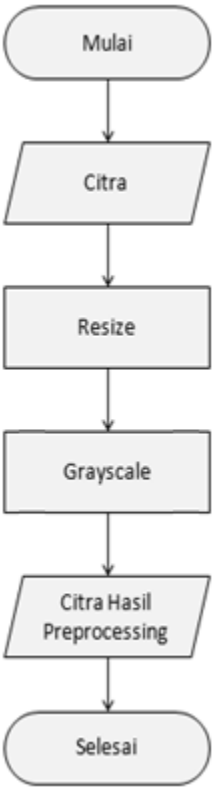
1. Perbaikan atau memodifikasi citra perlu dilakukan untuk meningkatkan kualitas penampakan atau untuk menonjolkan beberapa aspek informasi yang terkandung di dalam citra,
2. Elemen di dalam citra perlu dikelompokkan, dicocokkan, atau diukur,
3. Sebagian citra perlu digabung dengan bagian citra yang lain.

Pengolahan Citra bertujuan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (dalam hal ini komputer). Teknik-teknik pengolahan citra mentransformasikan citra menjadi citra lain. Jadi, masukannya adalah citra dan keluarannya juga citra, namun citra keluaran mempunyai kualitas lebih baik daripada citra masukan[10].

2.4.1 Preprocessing

Preprocessing adalah proses untuk memudahkan dalam menghasilkan nilai dari ekstraksi ciri. Pada tahap awal ini citra inputan akan dilakukan proses *resize* dan *grayscale*.

Berikut ini gambar dari proses *preprocessing* :

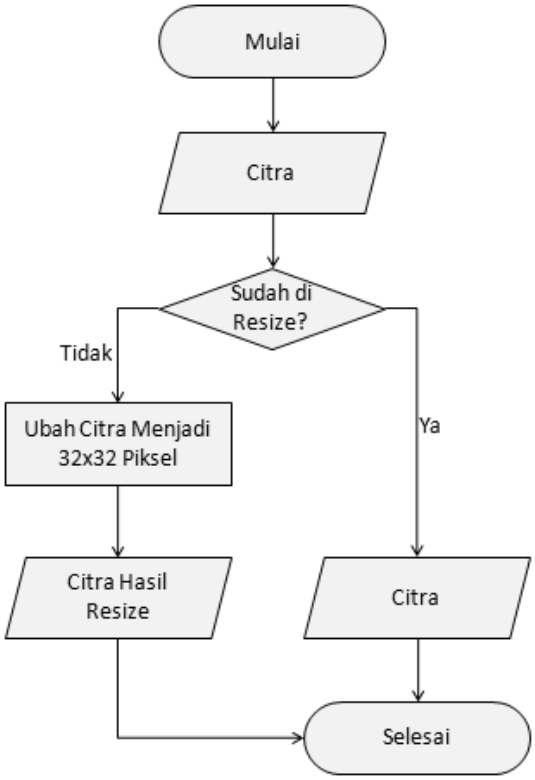


Gambar 2. 6 Proses Preprocessing

2.4.1.1 Resize

Pada proses preprocessing dilakukan proses *resize* citra. *Resize* adalah proses mengubah resolusi atau ukuran horizontal dan vertikal pada suatu citra, hal ini perlu dilakukan untuk menghasilkan ukuran yang sama pada semua citra yang dimasukkan. Tujuan dari *resize* citra adalah untuk mempercepat proses ekstraksi fitur warna dan fitur tekstur pada gambar yang memiliki dimensi (ukuran) yang besar [14].

Berikut ini gambar dari proses *resize* :

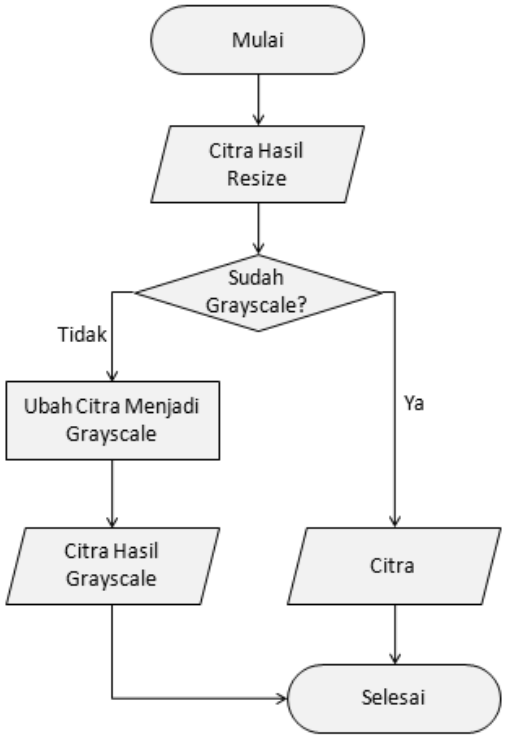


Gambar 2. 7 Proses Resize

2.4.1.2 Grayscale

Grayscale atau abu-abu pada sebuah image digital merupakan proses untuk mengubah warna dimana setiap pikselnya hanya berisikan informasi intensitas warna hitam dan putih. Untuk mendapatkan nilai graylevel pada citra maka dilakukan proses grayscale. Grayscale berfungsi untuk memudahkan proses klasifikasi citra agar lebih terfokus terhadap masing-masing piksel [15].

Berikut ini gambar dari proses g*rayscale* :



Gambar 2. 8 Proses Grayscale

2.5 Elemen Citra

Dalam pengolahan citra objek yang digunakan adalah sebuah citra dari sebuah objek tertentu yang mana citra tersebut mengandung sejumlah elemen dasar. Elemen dasar tersebut di manipulasi dalam pengolahan citra, elemen tersebut adalah:

1. Warna

   Warna adalah persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh objek. Setiap warna mempunyai panjang gelombang. Warna yang diterima oleh mata merupakan hasil kombinasi cahaya dengan panjang gelombang berbeda. Kombinasi warna yang memberikan rentang warna yang paling lebar adalah *red* (R), *green* (G), *blue* (B).

2. Kecerahan (*brightness*)

   Kecerahan disebut juga intensitas cahaya. Kecerahan pada sebuah piksel (titik) didalam citra bukanlah intensitas yang *rell*, tetapi sebenarnya adalah intensitas rata-rata dari suatu area yang melingkupinya.

3. Kontras (*contrast*)

   Kontras menyatakan sebaran terang dan gelap di dalam sebuah gambar. Citra dengan kontras rendah dicirikan oleh sebagian besar komposisi citranya adalah terang atau sebagian besar gelap. Pada citra dengan kontras yang baik, komposisi gelap dan terang tersebar secara merata.

4. Kontur (*contour*)

   Kontur adalah keadaan yang ditimbulkan oleh perubahan intensitas pada piksel yang bertetangga. Karena adanya perubahan intensitas, mata manusia dapat mendeteksi tepi objek didalam citra.

5. Bentuk (*shape*)

   Bentuk adalah properti intrinsik dari objek tiga dimensi, dengan pengertian bahwa *shape* merupakan properti intrinsik utama untuk sistem visual manusia. Pada umumnya citra yang dibentuk oleh mata merupakan citra *dwimatra* (dua dimensi), sedangkan objek yang dilihat umumnya berbentuk *trimatra* (tiga dimensi). Informasi bentuk objek dapat diekstraksi dari citra pada permulaan pra-pengolahan dan segmentasi citra.

6. Tekstur (*texture*)

   Tekstur diartikan sebagai distribusi spasial dari derajat keabuan di dalam sekumpulan piksel-piksel yang bertetangga. Jadi tekstur tidak dapat didefinisikan untuk sebuah piksel. Sistem visual manusia menerima informasi citra sebagai suatu kesatuan. Resolusi citra yang diamati ditentukan oleh skala dimana tekstur tersebut dipersepsi.

7. Waktu dan Pergerakan

   Respon suatu sistem visual tidak hanya berlaku pada faktor ruang, tetapi juga pada faktor waktu. Sebagai contoh, bila citra diam ditampilkan secara cepat, akan berkesan melihat citra yang bergerak.

8. Deteksi dan Pengenalan

   Dalam mendeteksi dan mengenali suatu citra, ternyata tidak hanya sistem visual manusia saja yang bekerja, tetapi juga ikut melibatkan ingatan dan daya pikir manusia.

2.6 Citra analog dan Citra Digital

Secara umum terdapat 2 jenis citra yaitu citra analog dan citra digital. Citra analog adalah citra yang bersifat kontinu, seperti gambar pada monitor televisi, foto sinar-X, foto yang tercetak dikertas foto, lukisan, pemandangan, hasil CT scan, gambar-gambar yang terekam pada pita kaset, dan lain-lain sebagainya.

Citra analog tidak dapat direpresentasikan dalam komputer sehingga tidak dapat diproses dikomputer secara langsung. Oleh sebab itu, agar citra ini dapat diproses dikomputer, proses konversi analog ke digital harus dilakukan terlebih dahulu. Citra analog dihasilkan dari alat-alat analog, *video* kamera analog, kamera foto analog, *Web Cam, CT scan,* sensor *ultrasound* pada system USG, dan lain-lain .

Citra digital adalah citra yang dapat diolah oleh komputer dan citra digital yaitu gambar pada bidang dua dimensi. Dalam tinjauan matematis, citra merupakan fungsi kontinu dari intensitas cahaya pada bidang dua dimensi. Ketika sumber cahaya menerangi objek, objek memantulkan kembali sebagian cahaya tersebut. Pantulan ini ditangkap oleh alat-alat pengindera optik, misalnya mata manusia, kamera, scanner dan sebagainya. Bayangan objek tersebut akan terekam sesuai intensitas pantulan cahaya. Ketika alat optik yang merekam pantulan cahaya itu merupakan mesin digital, misalnya kamera digital, maka citra yang dihasilkan merupakan citra digital. Pada citra digital, kontinuitas intensitas cahaya dikuantisasi sesuai resolusi alat perekam.

2.7 Tekstur

Secara umum tekstur mengacu pada repetisi elemen-elemen tekstur dasar yang sering disebut primitif atau *texel* (*texture element*). Suatu *texel* terdiri dari beberapa *pixel* dengan aturan posisi bersifat periodik, kuasiperiodik, atau acak.

Pengertian dari tekstur dalam hal ini adalah keteraturan pola-pola tertentu yang terbentuk dari susunan piksel-piksel dalam citra digital [9].

Untuk membentuk suatu tekstur setidaknya ada dua persyaratan yang harus dipenuhi antara lain :

1. Terdiri dari satu atau lebih piksel yang membentuk pola-pola primitif (bagian-bagian terkecil). Bentuk-bentuk pola primitif ini dapat berupa titik, garis lurus, garis lengkung, luasan dan lain-lain yang merupakan elemen dasar dari sebuah bentuk.
2. Munculnya pola-pola primitif yang berulang-ulang dengan interval jarak dan arah tertentu sehingga dapat diprediksi atau ditemukan karakteristik perulangannya.

Suatu citra memberikan interpretasi tekstur yang berbeda apabila dilihat dengan jarak dan sudut yang berbeda, manusia memandang tekstur berdasarkan deskripsi yang bersifat acak, seperti halus, kasar, teratur, tidak teratur, dan lain sebgainya. Hal ini merupakan deskripsi yang tidak tepat dan non-kuantitatif, sehingga diperlukan adanya suatu deskripsi yang kuantitatif (matematis) untuk memudahkan analisis [9].

2.7.1 Analisis Tekstur

Analisis tekstur merupakan dasar dari berbagai macam aplikasi, aplikasi dari analisis tekstur antara lain adalah penginderaan jarak jauh, pencitraan medis, identifikasi kualitas suatu bahan (kayu, kulit, tekstil dan lain-lain). Pada analisis citra, dikategorikan menjadi lima kategori utama yaitu statistis, struktural, geometri, model dasar dan pengolahan sinyal. Pendekatan statistis mempertimbangakan bahwa internsitas dibangkitkan oleh medan acak dua dimensi, metode ini berdasar pada frekuensi-frekuensi ruang. Contoh metode statistis adalah fungsi autokorelasi,run-length, matriks kookurensi, tranformasi fourier, frekuensi tepi. Teknik struktural berkaitan dengan penyusupan bagian-bagian terkecil suatu citra. contoh metode struktural adalah model fraktal. Metode geometri berdasar atas perangkat geometri yang ada pada elemen tekstur. Contoh model dasar adalah medan acak. Sedangkan metode pengolahan sinyal adalah

metode yang berdasarkan analisis frekuensi seperti transformasi gabor dan transformasi wavelet [9].

2.8 Metode Run Length

Grey level run length matrix yang biasa disingkat dengan GLRLM merupakan salah satu metode yang populer untuk mengekstrak tekstur sehingga diperoleh ciri statistik atau atribut yang terdapat dalam tekstur dengan mengestimasi piksel-piksel yang memiliki derajat keabuan yang sama. Ekstraksi tekstur dengan metode run-length dilakukan dengan membuat rangkaian pasangan nilai (*i,j*) pada setiap baris piksel. Perlu diketahui maksud dari run length adalah jumlah piksel berurutan dalam arah tertentu yang memiliki derajat keabuan/nilai intensitas yang sama. Jika diketahui sebuah matriksrun-lengthdengan elemen matriks*q* (*i, j* | θ) dimana i adalah derajat keabuan pada masing-masing piksel, j adalah nilai run-length, dan θ adalah orientasi arah pergeseran tertentu yang dinyatakan dalam derajat. Orientasi dibentuk dengan empat arah pergeseran dengan interval 45^0, yaitu 0^0, 45^0, 90^0, dan 135^0.

Berdasarkan penelitian yang dilakukan oleh Galloway[11], terdapat beberapa jenis ciri tekstural yang dapat diekstraksi dari matriks run-length. Berikut variabel-variabel yang terdapat di dari ekstraksi citra dengan menggunakan metode statistikal *Grey Level Run Length Matrix* :

i = nilai derajat keabuan

j = piksel yang berurutan (*run*)

M = Jumlah derajat keabuan pada sebuah gambar

N = Jumlah piksel berurutan pada sebuah gambar

r(j) = Jumlah piksel berurutan berdasarkan banyak urutannya (*run length)*

g(i) = Jumlah piksel berurutan berdasarkan nilai derajat keabuannya

s = Jumlah total nilai *run* yang dihasilkan pada arah tertentu

p(i,j) = himpunan matrik i dan j

n = jumlah baris * jumlah kolom.

Dimana variabel-variabel tersebut akan digunakan untuk mencari nilai dari atribut-atribut tekstur sebagai berikut [17]:

1. *Short Run Emphasis* (SRE)

   SRE mengukur distribusi short run. SRE sangat tergantung pada banyaknya short run dan diharapkan bernilai besar pada tekstur halus.

$$\mathbf{SRE} = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{P(i,j)}{j^2} \qquad (2.1)$$

2. *Long Run Emphasis* (LRE)

   LRE mengukur distribusi long run. LRE sangat bergantung pada banyaknya long run dan diharapkan bernilai besar pada tekstur kasar.

$$\mathbf{LRE} = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} P(i,j) * j^2 \qquad (2.2)$$

3. *Grey Level Uniformity* (GLU)

   GLU mengukur persamaan nilai derajat keabuan seluruh citra dan diharapkan bernilai kecil jika nilai derajat keabuan serupa diseluruh citra.

$$\mathbf{GLU} = \frac{1}{n_r} \sum_{i=1}^{N} \left(\sum_{j=1}^{M} P(i,j) \right)^2 \qquad (2.3)$$

4. *Run Length Uniformity* (RLU)

   RLU mengukur persamaan panjangnya run diseluruh citra dan diharapkan bernilai kecil jika panjangnya run serupa diseluruh citra.

$$\mathbf{RLU} = \frac{1}{n_r} \sum_{j=1}^{M} \left(\sum_{i=1}^{N} P(i,j) \right)^2 \qquad (2.4)$$

5. *Run Percentage* (RPC)

   RPC mengukur kebersamaan dan distribusi run dari sebuah citra pada arah tertentu. RPC bernilai paling besar jika panjangnya run adalah 1 untuk semua derajat keabuan pada arah tertentu.

$$\mathbf{RPC} = \sum_{j=1}^{M} \sum_{i=1}^{N} \frac{P(i,j)}{n} \qquad (2.5)$$

2.9 Klasifikasi Naïve Bayes

Naïve bayes adalah teknik prediksi berbasis probabilistik sederhana yang berdasar pada penerapan teorema Bayes[12].

1. Asumsi independensi (ketidaktergantungan) yang kuat (naif).
2. Model yang digunakan adalah "model fitur independen".

Independensi yang kuat pada fitur adalah bahwa sebuah fitur pada sebuah data tidak ada kaitannyadengan adanya atau tidak adanya fitur yang lain dalam data yang sama. Contoh: kasus klasifikasi hewan dengan fitur: penutup kulit, melahirkan, berat, dan menyusui.

1. Dalam dunia nyata, hewan yang berkembang biak dengan cara melahirkan dipastikan hewan tersebut menyusui juga, disini ada ketergantungan pada fitur menyusui karena hewan yang menyusui biasanya melahirkan, atau hewan yang bertelur biasanya tidak menyusui.
2. Dalam Bayes, hal tersebut tidak dipandang, sehingga masing-masing fitur seolah tidak ada hubungan apa-apa.

Kasus lain: prediksi hujan

1. Hujan tergantung angin, cuaca kemarin, kelembaban udara (tidak ada kaitan satu sama lain).
2. Tapi juga tidak boleh memasukkan fitur lain yang tidak ada hubungannya dengan hujan, seperti: gempa bumi, kebakaran, dsb.

Naïve bayes adalah suatu metode pengklasifikasian paling sederhana dengan menggunakan peluang yang ada, dimana diasumsikan bahwa setiap variable X

bersifat bebas (*independence*). Karena asumsi variabel tidak saling terikat, maka didapatkan :

$$P(X \mid Y = y) = \prod_{i=1}^{q} P(X_i \mid Y = y) \quad (2.6)$$

Data yang digunakan dapat bersifat kategorial (diskrit) maupun *continue*. Namun, pada tugas akhir ini akan digunakan data *continue*, karena hasil ekstraksi ciri citra merupakan data *continue* berupa angka angka hasil pengukuran tingkat *Short Run Emphasis* (SRE), *Long Run Emphasis* (LRE), *Grey Level Uniformity* (GLU), *Run Length Uniformity* (RLU), dan *Run Percentage* (RPC) pada ekstraksi ciri. Maka dari itu untuk data *continue* dapat diselesaikan dengan menggunakan langkah-langkah berikut.

Training :

1. Hitung rata-rata (*mean*) tiap fitur dalam dataset training dengan.

   $$\mu = \frac{\sum x_i}{n} \quad (2.7)$$

   Dimana:

   μ = mean

   n = banyaknya data

   $\sum x_i$ = jumlah nilai data

2. Kemudian hitung nilai varian dari diat dataset training tersebut seperti pada.

   $$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2 \quad (2.8)$$

   Dimana:

   σ^2= varians

   μ= mean

   x_i = nilai data

n = banyaknya data

Testing :

1. Hitung probabilitas (*Prior*) tiap kelas yang ada dengan cara menghitung jumlah data tiap kelas dibagi jumlah total data secara keseluruhan.

2. Selanjutnya menghitung densitas probabilitasnya. Fungsi densitas mengekspresikan probabilitas relatif. Data dengan mean μ dan standar deviasi σ, fungsi densitas probabilitasnya adalah :

$$\varphi_{\mu\sigma}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (2.9)$$

Dimana :

x = data masukan

π = 3,14

σ = standar deviasi

μ = mean

3. Setelah didapatkan nilai densitas probabilitasnya, selanjutnya menghitung posterior masing-masing kelas dengan menggunakan persamaan 2.10.

$$posterior = \frac{prior\ x\ likehood}{evidnce} \quad (2.10)$$

Atau bisa ditulis

$$Posterior = \frac{P(C_i)x\ P(X1 \dots\dots\dots\dots\dots\dots\dots Xn|C_i)}{P(X1 \dots\dots\dots\dots\dots\dots\dots Xn|C_i)} \quad (2.11)$$

Setelah didapat nilai posterior, kelas yang sesuai adalah nilai posterior terbesar.

2.10 Pengujian Black Box

Pengujian *black box* merupakan pendekatan komplementer dari teknik *white box*, karena pengujian *black box* diharapkan dapat mengetahui kesalahan kelas yang lebih luas dibandingkan teknik *white box*. Pengujian black box terfokus pada pengujian persyaratan fungsional perangkat lunak untuk menghasilkan serangkaian kondisi input yang sesuai dengan persayaratan fungsional suatu program.

Pengujian black box berusaha untuk menemukan kesalahan dalam kategori:

1. Fungsi yang hilang atau tidak tepat
2. Kesalahan interface
3. Kesalahan dalam struktur data
4. Kesalahan kinerja
5. Inisialisasi dan kesalahan terminasi

2.11 Pengujian Confusion Matrix

Confusion matrix adalah sebuah tabel yang menyatakan jumlah data uji yang benar diklasifikasikan. Berikut ini contoh *confusion matrix* untuk klasifikasi biner:

Tabel 2. 1 Confusion Matrix Untuk Klasifikasi Biner

		Kelas Prediksi	
		1	0
Kelas Sebenarnya	1	TP	FN
	0	FP	TN

Keterangan:

True Positive (TP), yaitu jumlah dokumen dari kelas 1 yang benar dan diklasifikasikan sebagai kelas 1.

True Negative (TN), yaitu jumlah dokumen dari kelas 0 yang benar diklasifikasikan sebagai kelas 0.

False Positive (FP), yaitu jumlah dokumen dari kelas 0 yang salah diklasifikasikan sebagai kelas 1.

False Negative (FN), yaitu jumlah dokumen dari kelas 1 yang salah diklasifikasikan sebagai kelas 0.

Untuk menghitung akurasi digunakan persamaan [18]::

$$Akurasi = \frac{TP + TN}{TP + FN + FP + TN} \; x100\%$$

2.12 Object Oriented Programming (OOP)

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapann prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktifitas analisis beorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek [13].

Pada saat ini, metode berorientais objek banyak dipilih karena metodologi lama banyak menimbulkan masalah seperti adanya kesulitan pada saat mentranformasi hasil dari satu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur, jenis aplikasi yang dikembangkan saat ini berbeda dengan masa lalu. Aplikasi yang dikembangkan saat ini sangat beragam (aplikasi bisnis, *real-time, utility* dan sebagainya) dengan platform yang berbeda-beda, sehingga menimbulkan tuntutan kebutuhan metodologi pengembangan yang dapat mengakomodasi ke semua jenis aplikasi.

Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut :

a. Meningkatkan Produktivitas

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reuseable*).

b. Kecepatan Pengembangan

Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengkodean

c. Kemudahan Pemeliharaan

Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering diubah-ubah.

d. Adanya Konsistensi

Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

e. Meningkatkan Kualitas Perangkat Lunak

Karena adanya pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengambangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Berikut beberapa contoh bahasa pemograman yang mendukung pemrograman berorientasi objek adalah :

a. Smalltalk

Smalltalk adalah salah satu bahasa pemograman yang diekmbangkan untuk mendukung pemrograman beroirentasi objek.

b. Bahasa Pemrograman Eiffel

Eiffel merupakan bahsa pemrograman yang kembangkan untuk mendukung pemrograman berorientasi objek oleh Bertrand Meyer dan *compiler*.

c. Bahasa Pemrograman Web (PHP)

Php dibuat pertama kali oleh seorang perekayasa perangkat (software engineering) yang bernama Rasmus Lerdoff.

d. Bahasa Pemrograman C++

C++ merupakan pengembangan lebih lanjut dari bahasa pemrograman C untuk mendukung pemrograman berorientasi objek.

e. Bahasa Pemrograman Java

Java dikembangkan oleh perusahaan Sun Microsystem. Java menurut definisi dari Sun Microsystem adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *standalone* ataupun pada lingkungan jaringan.

2.13 Pemrograman JAVA

2.13.1 Sejarah JAVA

Pada tahun 1991 sekelompok insinyur Sun yang dipimpin Patrick Naughton dan James Gosling ingin merancang bahasa komputer untuk perangkat konsumer seperti *cable TV box* yg bernama "Green". Karena perangkat itu tidak mempunyai banyak memory, maka harus menggunakan bahasa yang berukuran kecil dan menghasilkan kode.

Karena orang-orang di proyek *Green* berbasis C++ bukan Pascal, maka kebanyakan sintaks yang diambil dari C++, serta mengadopsi orientasi objek bukan prosedural. Mulanya bahasa yang diciptakan diberi nama "*Oak"*, kemudian diganti *Java* karena telah ada bahasa pemrograman yang bernama sama.

Pada tahun 1995, Netscape memutuskan membuat browser yang dilengkapi dengan *Java*. Setelah itu diikuti oleh *IBM, Symantec, Insprise*, bahkan *Microsoft*. Setelah itu gaung *Java* mulai terdengar, berbagai industri mulai meliriknya. Dengan strategi terbukanya, banyak universitas di Amerika, Jepang dan Eropa yang mengubah pengenalan bahasa pemrograman komputer menjadi *Java*, dan meninggalkan C++. Beberapa teknologi bahasa *Java* diantaranya:

a. J2EE (Java 2 Enterprise Edition)

   J2EE adalah kelompok dari beberapa API (Application Programming Interface) dari Java dan teknologi selain Java. J2EE dirancang untuk membuat aplikasi yang rumit. J2EE sering dianggap sebagai *middleware* atau teknologi yang berjalan di server, namun sebenarnya J2EE tidak hanya terbatas untuk itu. Faktanya J2EE juga mencakup teknologi yang dapat digunakan di semua lapisan dari sebuah sistem informasi. Implementasi J2EE menyediakan kelas dasar dan API dari Java yang

mendukung pengembangan dan rutin standard untuk aplikasi client maupun server, termasuk aplikasi yang berjalan di web browser.

b. J2SE (Java 2 Standard Edition)

   J2SE adalah inti/dasar dari bahasa pemrograman Java. JDK (Java Development Kit) adalah salah satu tool dari J2SE untuk mengkompilasi dan menjalankan program Java. Tool J2SE yang salah satunya adalah JDK 1.5 merupakan tool open source dari *Sun Microsystem*.

c. J2ME (Java 2 Micro Edition)

   J2ME adalah lingkungan pengembangan yang dirancang untuk meletakkan perangkat lunak Java pada barang elektronik beserta perangkat pendukungnya. Pada J2ME, jika perangkat lunak berfungsi baik pada sebuah perangkat, maka belum tentu juga berfungsi baik pada perangkat yang lainnya. J2ME membawa Java ke dunia informasi, komunikasi, dan perangkat komputasi selain perangkat komputer desktop yang biasanya lebih kecil dibandingkan perangkat komputer desktop. J2ME biasa digunakan pada telepon selular, pager, *Personal Digital Assistants* (PDA) dan sejenisnya.

2.13.2 Konsep Pemrograman Berorientasi Objek pada JAVA

Pemrograman berorientasi objek atau *Object Oriented Programing* (OOP) adalah inti dari pemrograman Java. Semua program Java merupakan objek. Maka dari itu, sebelum memulai penulisan kode-kode program Java sebaiknya mengetahui terlebih dahulu dasar-dasar dari konsep yang terkandung dalam pemrograman berorientasi objek. Beberapa ciri dari pemrograman berorientasi objek adalah:

a. Abstraksi (*abstraction*)

   Abstraksi adalah penyembunyian kerumitan dari suatu proses. Sebagai contoh, orang tidak perlu berpikir bahwa mobil adalah kumpulan dari puluhan atau ribuan bagian-bagian kecil (mesin, kabel, baut, roda, rem, dsb). Orang hanya perlu berfikir bahwa mobil adalah sebuah objek yang telah memiliki prilaku spesifik, yang dapat digunakan sebagai alat

transportasi, sehingga mereka tinggal mengendarainya tanpa harus mengetahui kerumitan proses yang terdapat di dalam mobil tersebut. Ini artinya, si pembuat mobil telah menyembunyikan semua kerumitan-kerumitan proses yang terdapat di dalam mobil dan pengguna tidak perlu mengetahui bagaimana sistem mesin, transmisi, dan rem bekerja. Konsep seperti inilah yang dinamakan dengan abstraksi.

b. Pembungkusan (*encapsulation*)

Secara kode program, proses abstraksi dapat dilakukan dengan cara pembungkusan semua kode dan data yang berkaitan ke dalam satu entitas tunggal yang disebut dengan objek. Dengan kata lain, sebenarnya proses pembungkusan itu sendiri merupakan cara untuk melakukan abstraksi. Dalam pemrograman tradisional, proses semacam ini dinamakan penyembunyian informasi *(information hidding)*. Dalam melakukan pembungkusan kode dan data dalam Java, terdapat tiga tingkat akses, yaitu: *private, protected,* dan *public.*

c. Pewarisan (*inheritance*)

Model pemrograman berorientasi objek juga menerapkan konsep pewarisan, sama halnya seperti yang terjadi di kehidupan nyata. Sebagai contoh, seorang anak pasti akan mewarisi beberapa sifat yang dimiliki oleh orang tuanya. Begitu juga di dalamsebuah program, suatu kelas dapat diturunkan menjadi kelas-kelas baru lainnya yang akan mewarisi beberapa sifat dari kelas induknya. Sebagai contoh, apabila memiliki kelas A dalam program yang di buat, maka dapat membentuk kelas-kelas turunan dari kelas A (misalnya: kelas B) yang mewarisi sifat-sifat yang terkandung dalam kelas A. Dalam proses kelas turunan tersebut, maka dapat menambahkan sifat dan perilaku baru kedalam kelas B yang sebelumya tidak dimiliki dalam kelas A. Ini dinamakan dengan konsep pewarisan (*inheritance*). Dalam terminologi Java, kelas induk dinamakan dengan *super class* dan kelas turunan disebut dengan *sub class.*

d. Polimorfisme (*polimorphism*).

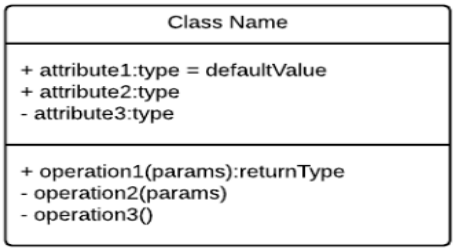
Polimorfisme adalah kemampuan suatu objek untuk mengungkap banyak hal yang melalui satu cara yang sama. Sebagai contoh, terdapat kelas A yang diturunkan menjadi kelas B, C, dan D. Dengan konsep polimorfisme, maka dapat menjalankan *method-method* yang terdapat pada kelas B, C, dan D hanya dari objek yang diinstansiasi dengan kelas A. Polimorfisme dinamakan dengan *dynamic binding, late binding,* maupun *runtime binding*.

2.14 Unified Modelling Language (UML)

Unified Modelling Language (UML) adalah sekumpulan spesifikasi yang dikeluarkan oleh OMG. UML terbaru adalah UML 2.3 yang terdiri dari 4 macam spesifikasi, yaitu : *Diagram Interchange Spesification,* UML *Infrastrukture,* UML *Superstrukture, dan Object Constraint Language* (OCL). Pada UML 2.3 terdiri 13 macam diagram yang dikelompokan pada 3 kategori, yaitu:

1. *Structure Diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
   a. Diagram Kelas

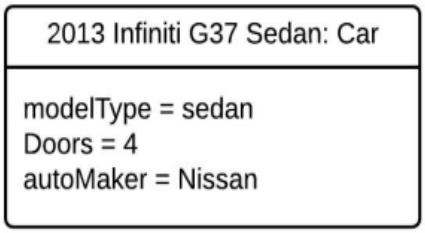
   Diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut attribut dan metode atau operasi.



Gambar 2. 9 Diagram Kelas

   b. Diagram Objek

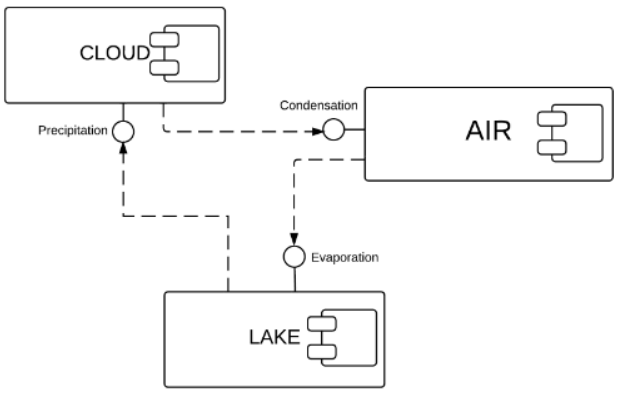
   Digram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem.



Gambar 2. 10 Diagram Objek

c. Diagram Komponen

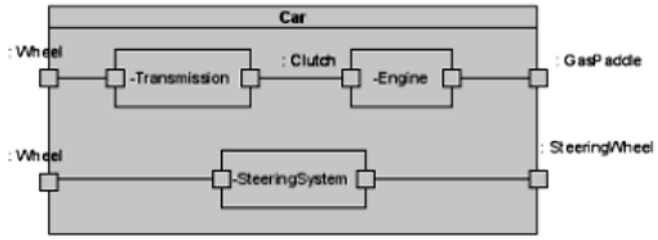
Diagram komponen dibuat untuk menunjukan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem.



Gambar 2. 11 Diagram Komponen

d. Composite Structure Diagram

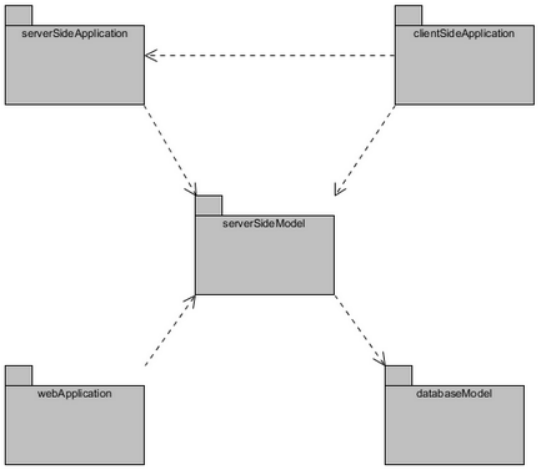
Composite Structure Diagram baru mulai ada pada UML versi 2.0. diagram ini dapat digunakan untuk menggambarkan struktur dari bagian-bagian yang saling terhubung maupun mendeskripsikan struktur pada saat berjalan (*runtime*).



Gambar 2. 12 Composite Structure Diagram

e. Package Diagram

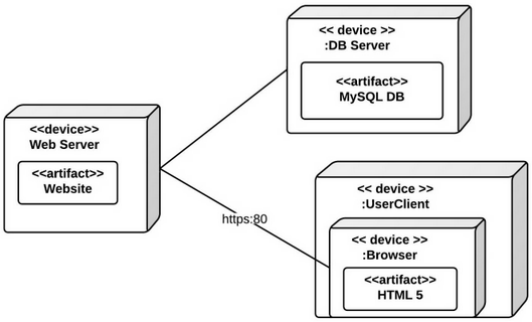
Package diagram menyediakan cara mengumpulkan elemen-elemen yang saling terkait dalam diagram UML. Hampir semua diagram dalam UML dapat dikelompokan menggunakan *Package Diagram*.



Gambar 2. 13 Package Diagram

f. Deployment Diagram

Deployment diagram menunjukan konfigurasi komponen dalam proses eksekusi aplikasi.

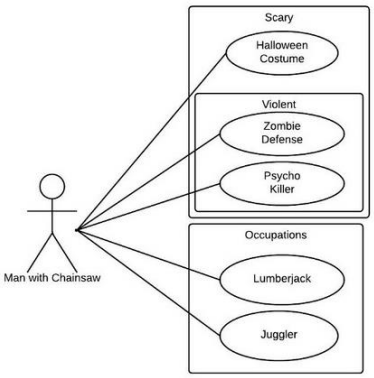


Gambar 2. 14 Deployment Diagram

2. *Behavior Diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. Pada *Behavior Diagram* dibagi menjadi 3 bagian :

a. Use Case Diagram

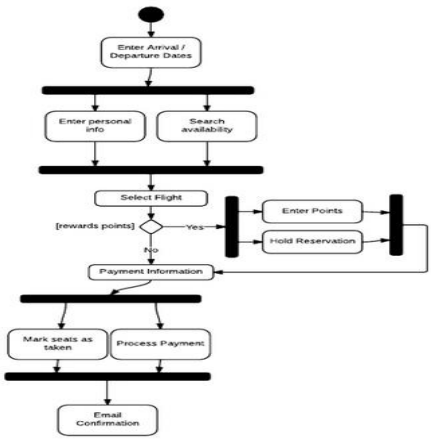
Use case diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat,. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat.



Gambar 2. 15 Use Case Diagram

b. Activity Diagram

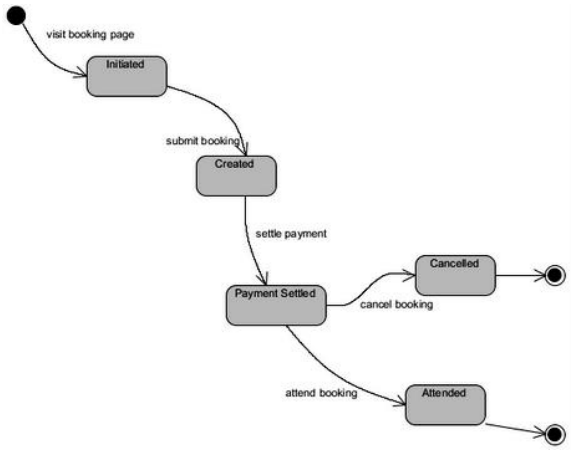
Activity diagram menggambarkan *workflow* atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak.



Gambar 2. 16 Activity Diagram

c. *State Machine Diagram*

State machine diagram digunakan untuk menggambarkan perubahan status atau transisi status dari sebuah mesin atau sistem atau objek.

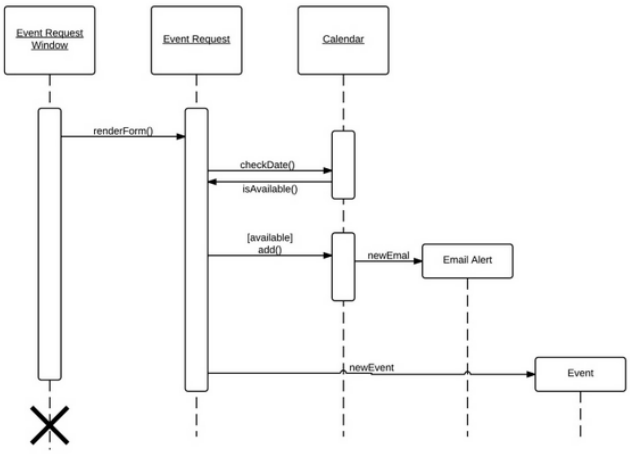


Gambar 2. 17 State Machine Diagram

3. *Interactions Diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi antar subsistem pada suatu sistem. Pada Interactions Diagram dibagi menjadi 4 bagian :

a. *Sequence Diagram*

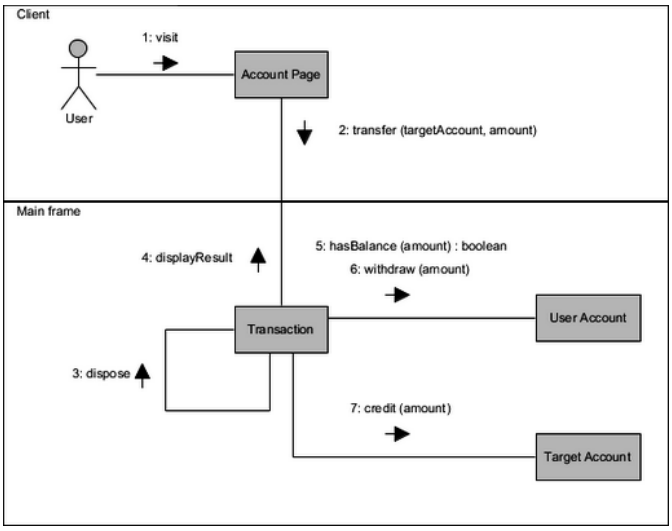
Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek.



Gambar 2. 18 Sequence Diagram

b. Communication Diagram

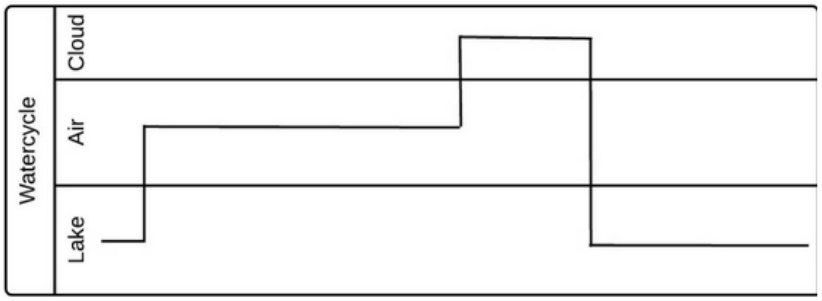
Communication diagram menggambarkan interaksi antar bojek/bagian dalam bentuk urutan pengiriman pesan. Diagram komunikasi merepresentasikan informasi yang diperoleh dari diagram kelas, diagram *sequence*, dan diagram *use case* untuk mendeskripsikan gabungan antara struktur statis dan tingkah laku dinamis dari suatu sistem.



Gambar 2. 19 Communication Diagram

c. Timing Diagram

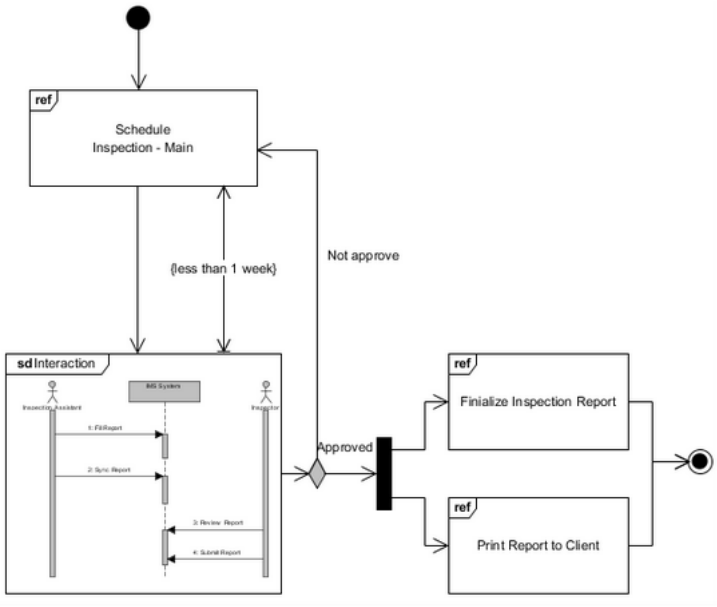
Timing diagram merupakan diagram yang fokus pada penggambaran terkait batas waktu.



Gambar 2. 20 Timing Diagram

d. Interaction Overview Diagram

Interaction overview diagram mirip dengan diagram aktivitas yang berfungsi untuk menggarbarkan sekumpulan urutan aktivitas, diagram ini adalah bentuk aktivias diagram yang setiap titik merepresentasikan diagram interaksi.



Gambar 2. 21 Interaction Overview Diagram