BAB 2
LANDASAN TEORI

1.1 Kecerdasan Buatan / AI()

Kecerdasan Buatan adalah bagian ilmu komputer yang membuat agar mesin komputer dapat melakukan pekerjaan seperti dan sebaik yang dilakukan manusia. Pada awal diciptakannya, komputer hanya difungsikan sebagai alat hitung saja. Namun seiring dengan perkembangan jaman, maka peran komputer semakin mendominasi kehidupan manusia. Komputer tidak lagi hanya digunakan sebagai alat hitung, lebih dari itu, komputer diharapkan untuk dapat diberdayakan untuk mengerjakan segala sesuatu yang bisa dikerjakan oleh manusia.

Manusia bisa menjadi pandai dalam menyelesaikan segala permasalahan di dunia ini karena manusia mempunyai pengetahuan dan pengalaman Pengetahuan diperoleh dari belajar. Semakin banyak bekal pengetahuan yang dimiliki oleh seseorang tentu saja diharapkan akan lebih mampu dalam menyelesaikan permasalahan. Namu bekal pengetahuan saja tidak cukup, manusia juga diberi akal untuk melakukan penalaran**,** mengambil kesimpulan berdasarkan pengetahuan dan pengalaman yang mereka miliki. Tanpa memiliki kemampuan untuk menalar dengan baik, manusia dengan segudang pengalaman dan pengetahuan tidak akan dapat menyelesaikan masalah dengan baik. Demikian pula dengan kemampuan menalar yang sangat baik, namun tanpa bekal pengetahuan dan pengalaman yang memadai, manusia juga tidak akan bisa menyelesaikan masalah dengan baik.

Agar komputer bisa bertindak seperti dan sebaik manusia, maka komputer juga harus diberi bekal pengetahuan dan mempunyai kemampuan untuk menalar. Untuk itu AI akan mencoba untuk memberikan beberapa metoda untuk membekali komputer dengan kedua komponen tersebut agar komputer bisa menjadi mesin pintar.[6]

1.1.1 Lingkup utama kecerdasan buatan

Lingkup utama kecerdasan buatan antara lain[6] :

1. Sistem pakar. Komputer digunakan sebagai saran untuk menyimpan pengetahuan para pakar. Dengan demikian komputer akan memiliki keahlian untuk menyelesaikan masalah dengan meniru keahlian yang dimiliki para pakar

2. Pengolahan bahasa alami. Dengan pengolahan bahasa alami ini diharapkan user mampu berkomunikasi dengan komputer dengan menggunakan bahasa sehari-hari.
3. Pengenalan ucapan. Melalui pengenalan ucapan diharapkan manusia mampu berkomunikasi dengan komputer dengan menggunakan suara.
4. Robotika dan Sistem sensor.
5. *Computer vision*, mencoba untuk dapat mengintrepetasikan gambar atau objek-objek tampak melalui komputer.
6. *Intelligent Computer aid Instruction*. Komputer dapat digunakan sebagai tutor yang dapat melatih dan mengajar.
7. *Game Playing*

1.1.2 Keuntungan Kecerdasan Buatan

Keuntungan kecerdasan buatan antara lain[7] :

1. Kecerdasan buatan lebih bersifat permanen. Kecerdasan alami akan cepat mengalami perubahan. Hal ini dimungkinkan karena sifat manusia yang pelupa. Kecerdasan buatan tidak akan berubah sepanjang sistem komputer dan program tidak mengubahnya.

2. Kecerdasan buatan lebih mudah diduplikasi dan disebarkan. Mentransfer pengetahuan manusia dari satu orang ke orang lain butuh proses dan waktu lama. Disamping itu suatu keahlian tidak akan pernah bisa diduplikasi secara lengkap. Sedangkan jika pengetahuan terletak pada suatu sistem komputer, pengetahuan tersebuat dapat ditransfer atau disalin dengan mudah dan cepat dari satu komputer ke komputer lain

3. Kecerdasan buatan lebih murah dibanding dengan kecerdasan alami. Menyediakan layanan komputer akan lebih mudah dan lebih murah dibanding dengan harus mendatangkan seseorang untuk mengerjakan sejumlah pekerjaan dalam jangka waktu yang sangat lama.
4. Kecerdasan buatan bersifat konsisten. Hal ini disebabkan karena kecerdasan busatan adalah bagian dari teknologi komputer. Sedangkan kecerdasan alami senantiasa berubah-ubah.
5. Kecerdasan buatan dapat didokumentasikan. Keputusan yang dibuat komputer dapat didokumentasikan dengan mudah dengan melacak setiap aktivitas dari sistem tersebut. Kecerdasan alami sangat sulit untuk direproduksi.
6. Kecerdasan buatan dapat mengerjakan pekerjaan lebih cepat dibanding dengan kecerdasan alami
7. Kecerdasan buatan dapat mengerjakan pekerjaan lebih baik dibanding dengan kecerdasan alami.

1.2 Swarm Intelligence

Swarm Intelligence adalah salah satu teknik kecerdasan buatan yang berlandaskan kepada perilaku kolektif (collective behaviour) pada sistem yang terdesentralisasi dan dapat mengatur dirinya sendiri (self-organizing).[9]

Sistem yang memanfaatkan Swarm Intelligence biasanya merupakan sebuah populasi yang terdiri atas anggota berupa agen-agen yang sederhana, yang berinteraksi secara lokal dengan sesama anggota, dan juga berinteraksi dengan lingkungan. Walaupun pada umumnnya tidak ada struktur kendali secara terpusat (centralized) yang mendikte bagaimana masing-masing individu bertindak, namun interaksi secara lokal (di antara anggota) seringkali menuju pada pembentukan (emergence) perilaku global.[10]

1.2.1 Sifat Sifat Dasar Swarm Intelligence

Sifat sifat dari *Swarm Intelligence* diantaranya [9]:

1. Terdiri dari banyak individu
2. Individu yang relatif bersifat homogen(Baik yeng memiliki sifat yang sama maupun yang memiliki sifat atau tipe yang berbeda)
3. Interaksi antar individu didasari dari aturan perilaku sederhana yang memanfaatkan informasi lokal baik secara langsung atau melalui lingkungannya

Kemampuan setiap individu melakukan interaksi dengan sesama individu maupun dengan lingkungannya dalam *swarm intelligence* disebut dengan *self-organizes.* Ini menunjukkan bahwa setiap individu dapat mengatur dirinya sendiri.

1.2.2 Prinsip Dasar Swarm Intelligence

Prinsip dasar Swarm Intelligence : [10]

1. Prinsip Kedekatan : setiap individu harus dapat menyesuaikan ruang dan waktu dengan perhitungan sederhana
2. Prinsip kualitas : setiap individu harus dapat menanggapi faktor lingkungan
3. Prinsip respon Beragam : setiap individu tidak seharusnya melakukan kegiatannya pada jalur yang terlalu sempit
4. Prinsip stabilitas : populasi tidak harus mengubah modus perilakunya setiap kali perubahan lingkungan
5. Prinsip Adaptasi : populasi harus mampu mengubah modus perilaku ketika komputasi bernilai penting atau berharga

1.2.3 Algoritma Swarm Intelligence

Adapun beberapa contoh algoritma atau metode yang populer dari swarm intelligence diantaranya [10] :

a.) Particle Swarm Optimization (PSO)

b.) Ant Colony System (ACS)

c.) Artificial Bees Colony (ABC)

1.3 Algoritma *particle swarm optimization* (PSO)

Algoritma *particle swarm optimization* (PSO) merupakan algoritma yang dikembangkan oleh Kennedy dan R. Eberhart pada tahun 1995 yang terinspirasi dari perilaku pergerakan kawanan hewan seperti ikan *(school of fish),* hewan herbivora *(herd),* burung *(flock),* dan kawanan serangga seperti semut rayap dan lebah. Tiap objek hewan disederhanakan menjadi partikel. Algoritma ini menerapkan sifat masing masing individu dalam satu kelompok besar kemudian menggabungkan sifat sifat tersebut untuk menyelesaikan masalah.[1] Algoritma PSO adalah salah satu metode dari teknik kecerdasan buatan *swarm intelligence* yang berlandaskan perilaku kolektif (*collective beharviour*) dan dapat mengatur dirinya sendiri (*self-organizing*). Dalam *swarm intelligence* algoritma *particle swarm optimization* ini merupakan metoda yang digunakan untuk penentuan posisi partikel terbaik [3]. Konsep optimasi partikel terdiri dari kerumunan. Pada persamaan dari algoritma PSO dasar yang di perkenalkan oleh oleh Kennedy dan R.Eberhart setiap partikel memiliki nilai fitness yang dievaluasi oleh fungsi yang dioptimasi, selain itu parameter lain yang dimiliki oleh tiap partikel adalah posisi partikel, konstanta, kecepatan partikel, posisi terbaik partikel dan posisi global terbaik. dari beberapa parameter yang di gunakan diatas dapat membentuk update posisi dilakukan secara serempak di tiap tiap partikel. Sebelum update posisi setiap partikel terlebih dahulu melakukan update kecepatan.

update partikel dilakukan dengan menggunakan persamaan dasar PSO [1]

$$V_i(t+1) = V_i(t) + C_1 r_1 (Pbest_i - X_i(t)) + C_2 r_2(Gbest_i - X_i (t)) \quad (2.1)$$

Update posisi partikel dapat dilakukan dengan menggunakan persamaan [1]:

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (2.2)$$

Dimana :

V_i (t)	adalah kecepatan partikel ke (i) pada iterasi ke (t) [1]
X_i (t)	adalah Posisi partikel ke (i) pada iterasi ke (t) [1]
V	adalah *Velocity* (kecepatan partikel) [1]
X	adalah Koordinat partikel (posisi partikel) [1]
i	adalah Partikel ke (1,2,3 . . . . . N) [1]
C_1 dan C_2	adalah Faktor *cognitive* dan sosial nilai ini adalah nilai konstanta bernilai diantara (0 dan 1) [1]
r_1 dan r_2	adalah Bilangan acak antara 0 – 1[1]
Pbest	adalah Posisi terbaik yang dimiliki setiap partikel [1]
Gbest	adalah posisi partikel yang memiliki posisi paling baik dari partikel lainnya. [1]

1.3.1 Penyelesaian Dalam PSO

Dalam PSO untuk menyelesaikan masalah langkah langkahnya adalah sebagai berikut :

1. Inisialisasi populasi partikel

Langkah ini merupakan tahap awal dimana jumlah dan koordinat partikel / posisi partikel ditentukan terlebih dahulu.

2. Melakukan perhitungan nilai fitness dari setiap partikel

Untuk memperoleh nilai fitness dibutuhkan nilai posisi setiap partikel tersebut untuk dievaluasi oleh fungsi fitness.

3. Menentukan Pbest (*personal best*)

Pbest adalah posisi terbaik setiap partikel. Posisi ini merupakan posisi yang ditempati oleh partikel saat ini.

4. Menentukan Gbest (*Global best*)

Nilai Gbest merupakan nilai posisi partikel yang memiliki posisi paling baik dari partikel lainnya. Nilai Gbest ini didapat dengan membandingkan nilai fitness dari setiap partikel untuk kemudian koordinat posisi dari partikel yang memiliki fitness paling optimal tersebut akan dijadikan Gbest. Nilai fitness dikatakan optimal tergantung kasus yang dimiliki apakah maksimasi ataupun minimasi

5. Menghitung nilai *velocity* (V)

Untuk mendapatkan nilai V terbaru dapat digunakan persamaan (2.1) untuk perbaharui kecepatan

6. Menghitung nilai koordinat terbaru

Memasukan hasil perhitungan nilai V dari setiap partikel kedalam persamaan (2.2)

7. Mengecek posisi partikel apakah sudah mencapai titik tujuan atau belum

Nilai dikatakan konveregen jika posisi partikel mencapai posisi tujuan. Jika partikel belum mencapai titik tujuan maka langkah 2 diulangi dengan mengevaluasi nilai partikel saat ini kedalam fungsi tujuan.’

1.3.2 Kelebihan dan Kekurangan PSO

A. Kelebihan dari algoritma PSO [11]:
   1. Algoritma PSO didasari oleh *intelligence.* Hal ini dapat diterapkan baik kedalam penelitian ilmiah maupun teknik.
   2. Algoritma PSO tidak memiliki perhitungan mutasi yang tumpang tindih. Pencarian dapat dilakukan dengan kecepatan setiap partikel. partikel yang optimal dapat mengirim informasi sehingga partikel lain dapat cepat melakukan pencarian.
   3. Perhitungan dari algoritma PSO sangat sederhana dibandingkan dengan perhitungan algoritma yang berkembang lainnya.Algoritma PSO ini memiliki tingkat optimasi yang paling tinggi dan dapat diselesaikan dengan perhitungan yang sederhana.
   4. Algoritma PSO mengandung bilangan real

B. Kekurangan dari algoritma PSO [11]:
   1. Algoritma ini mudah mengalami optimasi yang parsial yang mengakibatkan kurang tepatnya pengaturan terhadap kecepatan dan tujuan.
   2. Algoritma ini tidak dapat bekerja diluar masalah optimasi.
   3. Algoritma ini tidak dapat bekerja diluar masalah sistem koordinat

1.4 Game

Game merupakan bentuk partisipatif, interaktif dan hiburan. Menonton televisi, membaca, dan pergi ke teater merupakan segala bentuk hiburan pasif. Sedangkan ketika seseorang bermain *game*, mereka terhibur dengan berpartisipasi secara aktif. *Game* ditempatkan pada sebuah dunia buatan yang diatur melalui aturan-aturan (*rules*). Aturan tersebut menentukan tindakan atau langkah yang pemain dapat dan tidak dapat lakukan dalam sebuah *game.* [12]

1.4.1 Klasifikasi Game

Game dapat diklasifikasikan menjadi beberapa jenis, yaitu berdasarkan *platform* yang digunakan, dimensi, dan *genre* dari *game* itu sendiri. [12]

1.4.1.1 Klasifikasi Game Berdasarkan Platform

Platform merupakan kombinasi spesifik dari komponen elektronik atau perangkat keras komputer dengan perangkat lunak yang memungkinkan *game* untuk beroperasi. Berdasarkan *platform* yang digunakan, *game* dapat dibagi menjadi beberapa jenis, diantaranya: [12]

1. **Arcade Games**

Arcade games, yaitu *game* yang biasanya memiliki box atau mesin yang memang khusus di desain untuk jenis *video game*s tertentu, bahkan memiliki fitur yang dapat membuat pemainnya lebih merasa masuk dan menikmati , seperti pistol, kursi khusus, sensor gerakan, sensor injakkan dan stir mobil. *Arcade games* biasanya berada di daerah/tempat khusus, sebagai contoh di Indonesia dikenal dengan sebutan ding-dong [12]

2. **PC Games**

PC games, yaitu *game* yang dimainkan menggunakan komputer pribadi [12]

3. Consolle Game

Console games, yaitu *game* yang dimainkan menggunakan konsol tertentu, seperti *Playstation* 2, *Playstation* 3, XBOX 360, dan *Nintendo Wii* [12]

4. Handheald Game

Handheld games, yaitu *game* yang dimainkan di konsol khusus *video game* yang dapat dibawa kemana-mana (*portable*), sebagai contoh *Nintendo DS* dan *Sony PSP* [12]

5. Mobile Game

Mobile games, yaitu *game* yang dapat dimainkan khusus untuk *mobile phone* atau PDA [12]

1.4.1.2 Klasifikasi Game Berdasarkan Dimensi

Dimensi adalah angka yang berhubungan dengan sifat metrik atau topologi dari suatu objek. Berdasarkan dimensi dari objek-objeknya, *game* dapat dibagi menjadi beberapa jenis, diantaranya: [12]

1. Game 2D

Game dua dimensi atau 2D merupakan suatu konsep dimana semua objek berada pada satu bidang datar. Gerakan pada *game* 2D dibatasi hanya horizontal dan vertikal atau secara koordinat gerakan pemain dibatasi hanya dapat bergerak pada sumbu X danY. Pada g*ame* 2D terdapat dua pergerakan kamera. Pertama adalah kamera statis dimana gambar latar (*backround*) dan tempat *game* 2D tidak bergerak sama sekali, contoh dalam jenis ini adalah tetris. Kedua adalah *Side Scrolling* dimana *game* yang kita mainkan mempunyai kamera yang dapat bergeser ke kanan atau ke kiri dengan kecepatan sesuai dengan gerakan dan kecepatan karakter yang kita mainkan/gerakkan pada *game* tersebut, contoh *game* yang termasuk pada jenis ini adalah *Super Mario Bross, Sonic* dan *Megaman.* [12]

2. *Game* 2.5D

Setelah *game* 2D, muncul *game* dengan tampilan 3D datar (3D *Plane*). *Game* seperti ini bukan 2D tapi tidak juga full 3D. Teori grafik seperti ini disebut dengan 2.5D atau *pseudo*-3D, sedangkan pada istilah *game* lebih dikenal dengan istilah *isometric*, *diametric* atau *trimetric projection.* Biasanya *gameplay* dari *game* 2.5D mirip 2D dimana kita hanya bisa bergerak secara horizontal dan vertikal, namun beberapa objek menggunakan teknik *rendering* secara 3D. Salah satu hal yang membuat *developer game* membuat *game* seperti ini karena pemrosesan 3D secara total memerlukan banyak waktu dan biaya untuk membuatnya[12]

3. Game 3D

Game 3D menggunakan tiga dimensional representasi geometris data (X, Y, Z) yang disimpan dalam komputer untuk keperluan perhitungan dan *rendering* gambar 2D. Dalam sebuah *game* 3D, pemain bisa melihat sebuah objek dari sudut 360°. Terdapat tiga dasar dalam pembuatan sebuah objek 3D, yaitu 3D *modeling*, 3D *rendering*, dan 3D *computer graphics software* .

Genre pada suatu *game* memperlihatkan pola umum tantangan dari *game* tersebut. Dengan perkembangan informasi seperti sekarang, *genre* dari *game* masih terus berkembang, sebagai contoh simulasi menari (*dance simulation*) yang diperkenalkan oleh desainer *game* dari Jepang. Namun secara umum *game* dapat dibagi menjadi beberapa jenis [12]

1.4.1.3 Klasifikasi Game Berdasarkan Genre

Genre pada suatu *game* memperlihatkan pola umum tantangan dari *game* tersebut. Dengan perkembangan informasi seperti sekarang, *genre* dari *game* masih terus berkembang, sebagai contoh simulasi menari (*dance simulation*) yang diperkenalkan oleh desainer *game* dari Jepang. Namun secara umum *game* dapat dibagi menjadi beberapa jenis berdasarkan *genre* yang diterapkannya, yaitu: [12]

A. Game Strategi

Asal-usul dari *game* strategi berasal dari *game* papan seperti catur dan *Othello*. Pada g*ame* strategi biasanya pemain dapat mengendalikan tidak hanya satu karakter, melainkan beberapa karakter dalam *game* tersebut dengan berbagai jenis tipe kemampuan, kendaraan, hingga pembuatan berbagai bangunan, pabrik dan pusat pelatihan tempur, tergantung dari tema ceritanya. G*ame* strategi dibagi

menjadi 2 bentuk utama, yaitu *classical turn-based strategy* dan *real-time strategy*[12]

B. Role Playing Game (RPG)

Game RPG merupakan salah satu *genre* yang dibuat dari *game* yang berasal dari kertas dan pena. Dalam *game* RPG terdapat dua elemen utama yang menjadi kunci suksesnya sebuah *game* RPG. Fitur yang pertama adalah cerita (*story*) dan fitur yang kedua adalah pembangunan karakter (*character development*), dimana karakter merupakan tokoh hayalan yang akan di perankan oleh *player*. adapun game dengan jenis RPG ini di bagi menjadi 2 genre yaitu : [12]

1. **Action – RPG**

   game dengan *genre* ini sama seperti *game* dengan *genre game* RPG (*role playing game*) pada umumnya hanya saja ditambahkan unsur *action* sehingga sedikit membutuhkan keahlian dan kegesitan dalam memainkannya
2. **Turn based RPG**

   Untuk *ganre game* RPG satu ini sistem pertarungan lebih cenderung ke strategi. Dimana kita harus memberikan perintah kepada karakter seperti *Attack, Magic, Skill* dan memilih musuh yang di serang dalam setiap pertarungan. kelebihan *Turn Based* RPG adalah kita di tuntut untuk membuat taktik agar dalam sebuah pertarungan karakter kita dapat memenangkan pertarungan dan mendapat *experience point*.

C. Game Olahraga

Game olahraga mempunyai tantangan yang tidak biasa bagi seorang desainer *game*. Tidak seperti *game-game* yang lain, dimana pemain memiliki sedikit pengetahuan mengenai dunianya, tetapi *game* olahraga meniru aturan pada olahraga di kehidupan nyata[12]

D. Simulasi Kendaraan

Pada simulasi kendaraan, pemain dihadapkan dalam suatu kondisi seolah-olah pemain mengemudikan atau menerbangkan sebuah kendaraan, secara nyata atau imajinasi. Pada simulasi kendaraan yang nyata, salah satu tujuan yang harus dicapai adalah kemiripan kendaraan, seperti karakteristik kinerja mesin (kecepatan dan manuver) dengan kendaraan yang sebenarnya. Namun, jika mendesain kendaraan imajinasi, kita bebas untuk membuat *driving experience* untuk pemain, tanpa harus terbatas oleh gravitasi, *G-forces*, kapasitas bensin, dan lainnya[12]

E. Game Petualangan

Game petualangan bukan merupakan sebuah kompetisi atau simulasi seperti *game* yang lainnya. G*ame* jenis ini tidak menawarkan proses untuk dikelola atau mengalahkan musuh melalui strategi dan taktik. *Game* petualangan merupakan cerita interaktif mengenai karakter yang dikontrol oleh pemain[12]

F. Game Puzzle

Game puzzle merupakan *game* yang bertujuan untuk memecahkan sebuah puzzle, terkadang tanpa menyatukan dengan jalan cerita atau tujuan yang lebih besar. *Game* jenis ini biasanya bervariasi pada satu tema saja. Untuk kesuksesan secara komersil, sebuah *game* puzzle haruslah memiliki tantangan, visual yang atraktif, dan disamping itu, nyaman untuk dimainkan[12]

G. Game Aksi

Games aksi merupakan sumber yang baik bagi elemen desain dari sebuah *game*. *Game* ini relatif sederhana karena membuat analisis dari *game* tersebut lebih mudah, jika dibandingkan dengan *genre* lain. Aturan dari sebuah *game action* menjelaskan dasar dari mekanika *game.* Mereka biasanya lebih simpel dalam aturan, karena kealamian dari *gameplay*nya. Intinya, *game action* merupakan *game* yang membutuhkan keterampilan seperti pengolahan informasi sensorik dan tindakan secara cepat. Hal ini memaksa pemain untuk membuat keputusan dan melakukan tanggapan pada kecepatan yang jauh lebih besar. [12]

1.5 Pemrograman Berorientasi Object (PBO)

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang

berisi data dan operasi yang diberlakukan terhadapnya. Metologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktifitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek dan pengujian berorientasi objek[5].

Dalam rekayasa perangkat lunak, konsep pendekatan berorientasi objek dapat diterapkan pada tahap analisis perancangan,pemrograman dan pengujian perangkat lunak. Ada berbagai teknik yang dapat digunakan pada masing-masing tahap tersebut, dengan aturan dan alat bantu pemodelan tersebut[5]

1.5.1 Konsep Dasar PBO

Berikut ini adalah beberapa konsep dasar yang harus dipahami tentangmetodologi berorientasi objek :[5]

1.5.1.1 Object

Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata,seperti benda, manusia, satuan, tempat, kejadian, struktur, statusatau hal-hal lain yang bersifat abstrak. Objek merupakan suatuentitas yang mampu menyimpan informasi (status) dan mempunyaioperasi (kelakuan) yang dapat diterapkan atau dapat berpengaruhpada status objek. Objek mempunyai siklus hidup yaitu diciptakan,dimanipulasi dan dihancurkan.[5]

1.5.1.2 Kelas (Class)

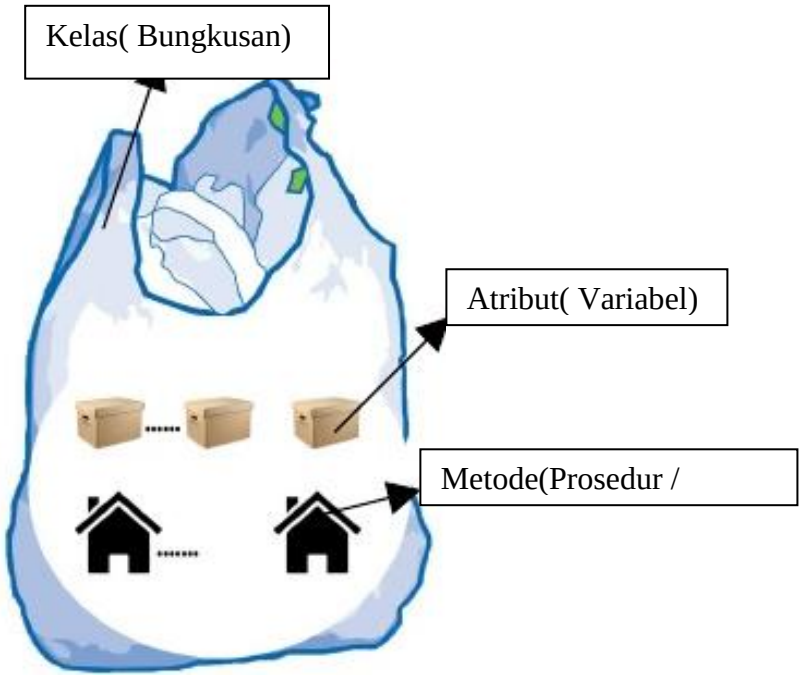
Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statis dan himpunan objek yang sama yang mungkin lahir atau diciptakan dan kelas tersebut.Sebuah kelas akan mempunyai sifat (atribut), kelakuan (operasi/metode), hubungan (relationship) dan arti. Suatu kelas dapat diturunkan dan kelas yang lain, di mana atribut dan kelas semula dapat diwariskan ke kelas yang baru.Secara teknis, kelas adalah sebuah struktur tertentu dalam pembuatan perangkat lunak. Kelas merupakan bentuk struktur padakode program yang menggunakan metodologi berorientasi objek [5]

1.5.1.3 Methode

Operasi atau metode atau method pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri.Operasi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek.Metode atau operasi dapat berasal dari event , aktivitas atau aksi keadaan, fungsi atau kelakuan dunia nyata. Contoh metode atau operasi misalnya :Read, Write, Move, Copy, dan sebagainya.[5]

1.5.1.4 Encapsulation

Pembukusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerja-nya. Misalkan seperti pada ilustrasi berikut. [5]



Gambar 1.1 Ilustrasi Enskapsulasi [5]

1.5.1.5 Inheritance (Pewarisan)

Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi objek lain sebagai bagian dari dirinya.[5]

1.5.1.6 Polimorfisme

Kemampuan suatu objek untuk digunakan di banyak tujuan yangberbeda dengan nama yang sama sehingga menghemat barisprogram. [5]

1.5.1.7 Antarmuka

Antarmuka atau interface sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi.Deklarasi metode pada sebuah interface dapat diimplementasikan oleh kelas lain.Sebuah kelas dapat mengimplementasikan lebih dari satu antarmuka di mana kelas ini akan mendeklarasikan metode pada antarmuka yang dibutuhkan oleh kelas itu sekaligus mendefinisikanisinya pada kode program itu. Metode pada antarmuka yang diimplementasikan pada suatu kelas harus sama persis dengan yang ada pada antarmuka . antarmuka atau interface Biasanya digunakan agar kelas yang lain tidak mengakses langsung ke suatu kelas, mengakses antarmukanya. [5]

1.5.1.8 Reusability

Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut. [5]

1.5.1.9 Generalisasi Spesialisai

Menunjukkan hubungan antar kelas dan objek yang umum dengankelas dan objek yang khusus. Misalnya kelas yang lebih umum(generalisasi) adalah kendaraan darat dan kelas khususnya(spesialisasi) adalah mobil, motor dan kereta. [5]

1.5.1.10 Komunikasi Antar Object

Komunikasi antar objek dilakukan lewat pesan (message) yang dikirim dari satu objek ke objek lainnya. [5]

1.5.1.11 Package

Package adalah sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam package yang berbeda. [5]

1.5.1.12 Atribute

Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai elemen-elemen data yang dimiliki oleh objek dalam kelas objek Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama dan sebagainya. [5]

1.5.1.13 Abstraksi

Prinsip untuk merepresentasikan dunia nyata yang kompleksmenjadi satu bentuk model yang sederhana dengan mengabaikanaspek-aspek lain yang tidak sesuai dengan permasalahan. [5]

1.5.2 Unified Modelling Language (UML)

Unified Modelling Language (disingkat UML) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan kebutuhan (requirement), membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek.[5]

UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataanya UML paling banyak digunakan pada metodologi berorientasi objek. Berikut merupakan macam-macam diagram UML:

1.5.2.1 Use Case Diagram

a. Menggambarkan bagaimana sistem digunakan
b. Mendeskripsikan apa yang sistem lakukan tanpa mendeskripsikan bagaimana sistem menyelesaikannya.
c. Dibuat berdasarkan interaksi dan relasi dari individual use case.
d. Berisi aktor, event, dan use case.
e. Titik awal untuk pemodelan UML.

1.5.2.2 Activity Diagram

Menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir.

1.5.2.3 Sequence Diagram

Menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainnya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagrams* terdiri atas dimensi vertical (waktu) dan dimensi horizontal (objek-objek yang terkait).

1.5.2.4 Class Diagram

Menggambarkan keadaan suatu sistem dengan menjelaskan keterhubungan antara suatu *class* dengan *class* yang lain yang terdapat pada sistem.