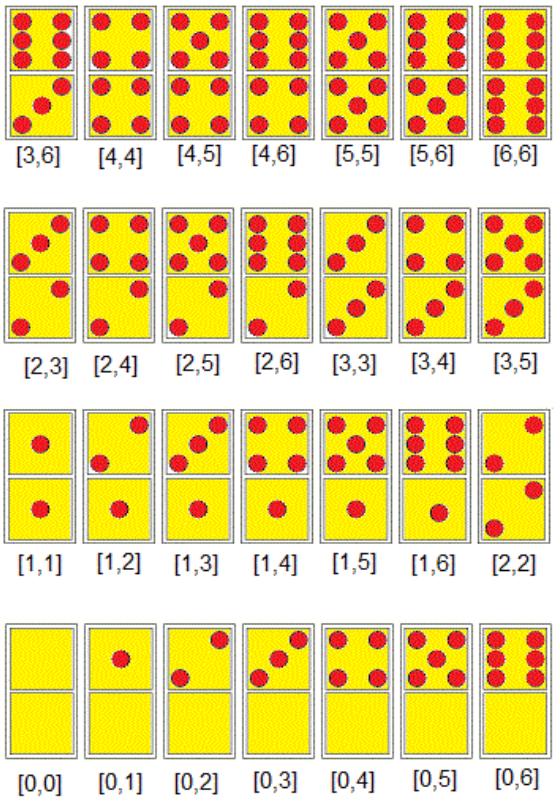
BAB II
LANDASAN TEORI

2.1 Domino

Domino merupakan sebuah permainan yang menggunakan balok-balok dimana pada kedua sisi memiliki nilai berupa tanda lubang berwarna hitam atau merah yang menyatakan nilainya dari 0 sampai dengan 6. Saat ini permainan domino ada yang menggunakan kertas dan tanda yang berbentuk bulat digunakan untuk menyatakan nilai dari kartu domino tersebut. Permainan domino umunya dimainkan oleh 4 orang. Jumlah kartu domino keseluruhan adalah 28 kartu seperti pada gambar 2.1 [1].



Gambar 2. 1. Domino

2.1.1 Sejarah Domino

Permainan domino adalah sebuah permainan kartu yang sudah sangat populer dikalangan warga Indonesia. Permainan ini biasanya dilakukan di saat santai atau saat berkumpul bersama teman-teman. Di beberapa daerah di Indonesia, domino dikenal juga dengan sebutan gaple atau permainan gaple.

Permainan ini di Indonesia lebih sering dimainkan masyarakat kalangan menengah ke bawah, padahal menurut catatan sejarah, saat pertama dibuat, domino merupakan persembahan seorang abdi dalem untuk sang kaisar. Hanya para bangsawan-lah yang saat itu bisa memainkan domino.

Seperti dikutip pagat.com, dalam sejarah domino menyebutkan permainan domino lahir di Cina. Menurut versi pertama menyebutkan bahwa domino mulai dikenal tahun 1120 Masehi. Versi kedua mengungkapkan bahwa domino mulai dimainkan oleh tentara Hung Ming pada sekitar tahun 181-234 Masehi. Namun sejarawan lain menyampaikan versi ketiga bahwa domino sudah dikenal pada masa Keung T’ai Kung, sekitar 1.100 tahun Sebelum Masehi (SM).

Lalu dari Cina, permainan ini menyebar hingga Eropa. Ahli sejarah lain bernama Michael Dummet menuliskan dalam buku "*Game of Tarot*," mengungkapkan bahwa di Eropa, domino mulai banyak dimainkan di Naples dan Venice pada awal abad ke-18.

Meski permainan domino berasal dari Cina, namun kata "domino" bukan berasal dari bahasa Cina. Bisa jadi, saat pertama dipersembahkan kepada kaisar Cina, permainan tersebut masih menggunakan nama yang lain. Kata domino berasal dari bahasa latin "dominus" yang berarti tuan rumah. Sementara dalam bahasa Skotlandia dan Inggris ada juga kata yang merujuk pada kata tersebut, yaitu "domini" yang berarti kepala sekolah.

2.1.2 Aturan Permainan Domino

Cara memainkan permainan domino adalah [1] :

1. Dengan meletakkan kartu domino yang bernilai paling kecil terlebih dahulu yaitu kartu kosong.
2. Diikuti oleh pemain lain dengan menyambung kartu domino dengan nilai yang bersesuaian sehingga membentuk suatu pola yang tidak terputus.
3. Penurunan kartu dilakukan hingga kartu domino habis atau tidak ada lagi kartu yang dapat diturunkan
4. Permainan berakhir jika terdapat pemain yang dominonya telah habis atau semua pemain tidak dapat lagi menyambungkan balok-balok itu lagi.
5. Pemain dengan nilai angka yang paling sedikit adalah pemenangnya.

2.2 Kecerdasan Buatan

Kecerdasan Buatan (*Artificial Intelligence*) di definisikan sebagai kecerdasan yang di tunjukkan oleh suatu entitas buatan. Sistem seperti ini umumnya dianggap komputer. Kecerdasan diciptakan dan dimasukkan ke dalam suatu mesin (Komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Salah satu macam bidang yang menggunakan kecerdasan buatan adalah permainan komputer (*games*) [5].

2.2.1 Tujuan Akhir AI (*Artificial Intelligence*)

Terdapat sembilan tujuan AI [5] yaitu:

1. Memahami kognisi manusia. Bagaimana manusia menyelesaikan masalah? Coba untuk mendapatkan pengetahuan ingatan manusia yang mendalam, kemampuan problem-solving, belajar, membuat keputusan, dan lain-lain.
2. Otomatisasi biaya-efektif. Menggantikan manusia dalam tugas-tugasi intelegensi.

3. Penguatan intelegensi biaya-efektif. Membangun sistem untuk membantu manusia berpikir lebih baik, lebih cepat, lebih dalam.
4. Intelegensi manusia super. Membangun program yang mempunyai kemampuan untuk melebihi intelegensi manusia.
5. Problem-solving umum. Sistem penyelesaian berbagai masalah yang luas.
6. Wacana koheren. Komunikasi dengan manusia mengguanakan bahasa, alami.
7. Otonomi. Mempunyai sistem intelegensi yang beraksi atas inisiatif sendiri.
8. Belajar (induksi). Sistem sebaiknya dapat untuk memperoleh data sendiri, dan tau bagaimana memperolehnya.
9. Informasi. Simpan informasi dan mengetahui bagaimana untuk mengambil informasi.

2.2.2 Karakteristik AI

Karakteristik ini didasarkan pada pandangan bahwa AI ada 4 kategori [5] yaitu:

1. Sistem yang dapat berpikir seperti manusia.
2. Sistem yang dapat beraksi seperti manusia.
3. Sistem yang dapat berpikir secara rasional.
4. Sistem yang dapat beraksi secara rasional.

2.3 Konsep Dasar Algoritma

Defenisi Algoritma, “Algoritma adalah urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis”. Sedangkan menurut Kamus Besar Bahasa Indonesia, terbitan Balai Pustaka, 1988. “Algoritma adalah urutan logis pengambilan putusan untuk pemecahan masalah” [6].

2.3.1 Dasar Teori *Minimax*

Algoritma *minimax* (juga sering disebut Minmax) adalah sebuah algoritma yang mendasari pola pikir langkah penyelesaian masalah dalam beberapa jenis permainan komputer, seperti tic-tac-toe, othello, checkers, catur. Pada dasarnya, algoritma *minimax* sangat andal untuk menyelesaikan segala masalah dalam pencarian langkah untuk permainan komputer dengan jumlah kemungkinan penyelesaian yang kecil, seperti pada permainan tic-tac-toe.

Secara teori, algoritma *minimax* didefinisikan sebagai berikut:

"Untuk setiap permainan satu-lawan satu, ada sebuah nilai yang bernilai V dan strategi yang dipilih oleh tiap pemain sedemikian sehingga dengan strategi dari pemain-2, pemain-1 dapat memperoleh nilai terbaik V dan sebaliknya dengan strategi dari pemain-1, pemain-2 dapat memperoleh nilai terbaik –V, sehingga: (a), Jika diberikan strategi dari pemain ke-2, maka langkah penyelesaian terbaik dari pemain pertama adalah V. Dan (b), jika diberikan strategi dari pemain pertama, maka langkah penyelesaian terbaik dari pemain kedua adalah –V". Singkatnya, pemain pertama memberikan langkah penyelesaian yang bernilai V terhadap permainan pemain kedua, dan sebaliknya, pemain kedua memberikan langkah peyelesaian bernilai –V. Pemikiran inilah yang mendasari asal usul penamaan algoritma *Minimax*, dimana pemain yang satu berjuang untuk mendapat nilai maksimum, sedangkan lawannya berjuang untuk mendapat nilai minimum [6].

Cara kerja algoritma *Minimax*:

1. Misalkan ada 2 pemain yang terlibat, kita namakan MAX dan MIN
2. Lakukan pencarian atau penulusuran dari simpul yang paling dalam
3. setelah simpul yang paling dalam ditelusuri akan dilakukan pencarian tingkat kedua, ketiga dan seterusnya .
4. Pencarian dilakukan secara rekursif
5. lakukan pembandingan nilai MAX dan MIN

6. Nilai MIN dan MAX yang telah dibandingkan akan disimpan pada temp sementara
7. Tentukan Nilai MAX untuk menentukan nilai Root.

2.3.2 Algoritma *Minimax*

Algoritma *minimax* merupakan salah satu algoritma yang digunakan pada permainan papan (*board game)* yang pada umumnya dimainkan oleh 2 orang pemain atau lebih dan berbasis zero-sum yang artinya keuntungan untuk pemain pertama berarti kerugian untuk pemain kedua.

Algoritma *minimax* akan melakukan pengecekan terhadap seluruh kemungkinan yang ada sampai permainan selesai. Keuntungan yang didapat dengan menggunakan algoritma *minimax* yaitu algoritma *minimax* mampu menganalisis segala kemungkinan posisi permainan untuk menghasilkan keputusan yang terbaik karena algoritma *Minimax* ini bekerja secara rekursif dengan mencari langkah yang akan membuat lawan mengalami kerugian minimum. Semua strategi lawan akan dihitung dengan algoritma yang sama dan seterusnya. Ini berarti, pada langkah pertama komputer akan menganalisis seluruh kondisi permainan, dan untuk setiap langkahnya, komputer akan memilih langkah yang paling membuat lawan mendapatkan keuntungan minimum, dan yang paling membuat komputer itu sendiri mendapat keuntungan maksimum. Dalam penentuan keputusan tersebut dibutuhkan suatu nilai yang merepresentasikan kerugian atau keuntungan yang akan diperoleh jika langkah tersebut dipilih [6]

2.3.3 Implementasi Algoritma *Minimax*

Misalkan ada sebuah permainan sederhana yang dengan hanya ada satu langkah, untuk tiap pemain dengan kemungkinan situasi seperti pada tabel 2.1.

Tabel 2. 1. Contoh nilai akhir pergerakan MAX dan MIN

Pergerakan MAX	Pergerakan MIN	Nilai Evaluasi
A	C	12
A	D	-2
B	C	5
B	D	6

Fungsi evaluasi berlaku untuk posisi akhir papan, yang berupa kombinasi langkah dari MIN dan MAX. MAX berasumsi jika MIN akan bermain dengan baik. Maka dari itu, MAX tahu jika dia melakukan pergerakan A, maka MIN akan membalasnya dengan melakukan pergerakan D, yang mengakibatkan nilai evaluasi -2 (kemenangan bagi MIN). Bagaimanapun juga, jika MAX melakukan gerakan B, dia pasti akan menang karena pergerakan MIN yang terbaik hanya akan menghasilkannilai evaluasi terbaik sebesar 5 saja. Jadi, denganmenggunakan algoritma *minimax*, MAX akan selalu memilih untuk melakukan langkah B, walaupun dia sebenarnya akan mendapatkan kemenangan yang lebih baik jika melakukan A dan MIN melakukan kesalahan dengan memilih langkah C [6].

2.3.4 Metode *Divide* and *Conquer*

Metode *Divide and Conquer* adalah metode pemecahan masalah dengan membagi masalah ke menjadi submasalah yang lebih kecil, memecahkan setiap submasalah secara rekursif dan menggabungkan hasil yang sesuai untuk sebuah solusi lengkap. *Divide* and *Conquer* memiliki tiga proses utama yaitu [7]:

a. Divide: membagi masalah menjadi beberapa submasalah yang memiliki kemiripan dengan masalah semula namun berukuran lebih kecil (idealnya berukuran hampir sama)
b. Conquer: memecahkan (menyelesaikan) masing-masing submasalah (secara rekursif)
c. Combine: mengabungkan solusi masing-masing submasalah sehingga membentuk solusi masalah semula.

Rumus yang digunakan pada metode *Divide* and *Conquer* adalah [7]:

$$K = (i+j) \text{ div } 2 \quad \ldots(2.1)$$

dimana :

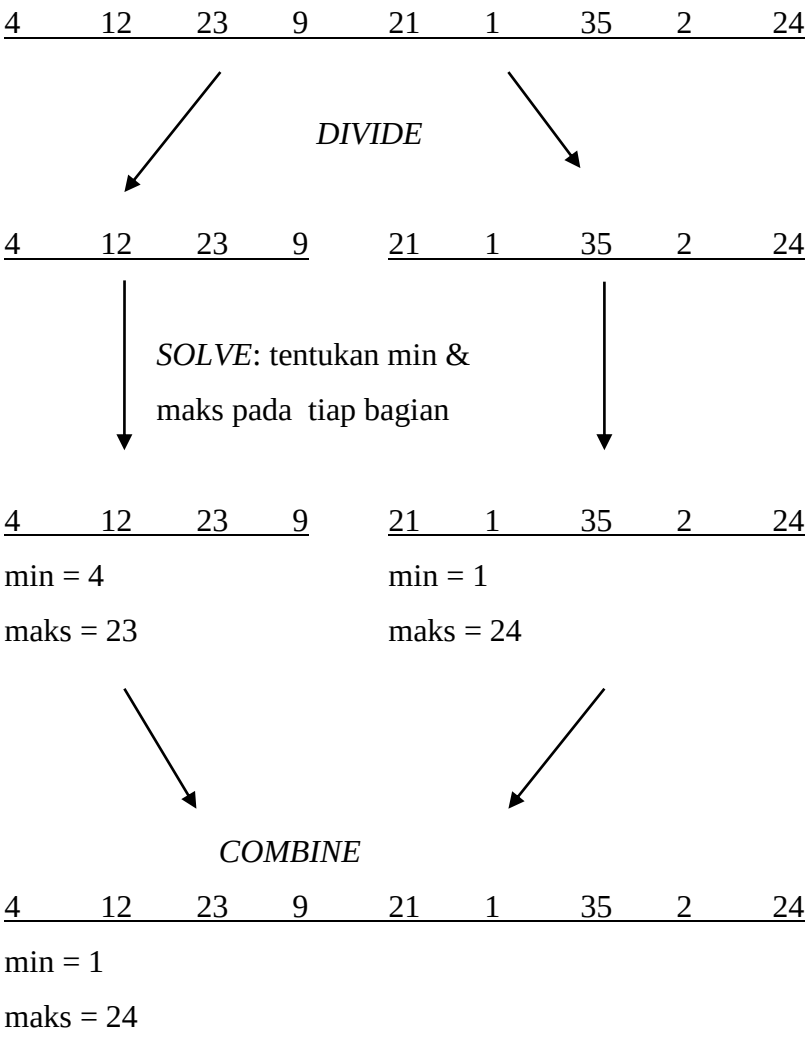
a. K = Pemanggilan
b. I = 1
c. J = N

Contoh Penyelesaian :

Misalkan tabel *A* berisi elemen-elemen sebagai berikut:

4 12 23 9 21 1 35 2 24

Ide dasar algoritma secara *Divide and Conquer*:



Ukuran tabel hasil pembagian dapat dibuat cukup kecil sehingga mencari minimum dan maksimum dapat diselesaikan (SOLVE) secara lebih mudah. Dalam hal ini, ukuran kecil yang dipilih adalah 1 elemen atau 2 elemen.

Algoritma MinMaks:

1. Untuk kasus $n = 1$ atau $n = 2$,

   SOLVE: Jika $n = 1$, maka $min = maks = A_n$.

   Jika $n = 2$, maka bandingkan kedua elemen untuk menentukan *min* dan *maks*.

2. Untuk kasus $n > 2$,

   a. *DIVIDE*:

   Bagi dua tabel *A* secara rekursif menjadi dua bagian yang berukuran sama, yaitu bagian kiri dan bagian kanan.

   b. *CONQUER*:

   Terapkan algoritma *Divide and Conquer* untuk masing-masing bagian, dalam hal ini *min* dan *maks* dari tabel bagian kiri dinyatakan dalam peubah *min1* dan *maks1*, dan *min* dan *maks* dari tabel bagian kanan dinyatakan dalam peubah *min2* dan *maks2*.

   c. *COMBINE*:

   Bandingkan *min1* dengan *min2* untuk menentukan *min* tabel *A*

   Bandingkan *maks1* dengan *maks2* untuk menentukan *maks* tabel *A*.

DIVIDE dan CONQUER: Bagi tabel menjadi dua bagian sempai berukuran 1 atau 2 elemen:

4	12	23	9	21	1	35	2	24	
4	12	23	9	21	1	35	2	24	
4	12	23	9	21	1	35	2	24	

SOLVE dan COMBINE: Tentukan min dan maks masing-masing bagian tabel, lalu gabung:

4 12	23 9	21 1	35	2 24
min = 4	min = 9	min = 1	min = 35	min = 2
maks = 12	maks = 23	maks = 21	maks =35	maks = 24

4 12 23 9	21 1	35 2 24
min = 4	min = 1	min = 2
maks = 23	maks = 21	maks = 35

4 12 23 9	21 1 35 2 24
min = 4	min = 1
maks = 23	maks = 35

4 12 23 9 21 1 5 2 24
min = 1
maks = 35

Jadi, nilai minimum tabel = 1 dan nilai maksimum = 35.

2.4 Pemrograman Beorientasi Objek

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 penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek.

Pada saat ini, metode berorientasi objek banyak dipilih karena metodologi lama banyak menimbulkan masalah seperti adanya kesulitan pada saat mentransformasi hasil dari suatu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur, jenis aplikasi yang dikembangkan saat ini berbeda dengan masa lalu. Aplikasi yang dikembangkan pada 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 tersebut [8].

Keuntungan menggunakan metode berorientasi objek adalah sebagai berikut [4]:

1. Meningkatkan produktivitas

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

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

   Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.
4. Adanya konsistensi

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.
5. Meningkatkan kualitas perangkat lunak

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

2.4.1 Konsep Dasar PBO

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem (sistem perangkat lunak, sistem informasi, atau sistem lainnya). Pendekatan berorientasi objek akan memandang sistem yang

akan dikembangkan sebagai suatu kumpulan onjek yang berkorespondensi dengan objek-objek dunia nyata.

Ada banyak cara untuk mengabstraksikan dan memodelkan objek-objek tersebut, mulai dari abstraksi objek, kelas, hubungan antar kelas sanpai sistem. Saat mengabstraksikan dan memodelkan objek, data dan proses-proses yang dipunyai oleh pihak objek akan dienkapsulasi (dibungkus) menjadi satu kesatuan [4].

Berikut ini adalah beberapa konsep dasar yang harus dipahami tentang metodolohi berorientasi objek:

a. Kelas (*class*)

   kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan onjek yang sama yang mungkin lahir atau diciptakan dan kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut). Kelakuan (operasi/metode), hubungan (relathionship) dan arti. Suatu kelas dapat diturunkan dan kelas yang lain, dimana atribut dan kelas semula dapat diwariskan ke kelas yang baru.

b. Object (*object*)

   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstark. Objek merupakan suatu entitas yang mampu menyimpan informasi dan mempunyai operasi yang dapat diterapkan atau dapat berpengaruh pasa status objeknya.

   Secara teknis, sebuah kelas saat program dieksekusi maka akan dibuat sebuah objek. Objek dilihat dari segi teknis adalah elemen pada saat runtime yang akan diciptakan, dimanipulasi, dan dihancurkan saat eksekusi sehingga sebuah objek hanya ada saat sebuah program dieksekusi, jika masih dalam bentuk kode, disebut sebagai kelas jadi pada saat runtime, yang kita punya adalah objek, didalam teks program yang kita lihat hanyalah kelas.

c. Metode.

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 berfungsi untuk memanipulasi objek itu sendiri. Operasi atau metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek.

d. Atribut (*attribute*)

Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya. Atribut sebaiknya bersifat privat untuk menjaga konsep enkapsulasi.

2.5 UML (Unified Modeling Language)

UML adalah bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek. UML banyak digunakan untuk mendefinisikan *requirement*, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. UML diperlukan karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung [4].

2.5.1 Kegunaan UML

UML merupakan bahasa pemodelan general-purpose yang evolusioner, mudah diaplikasikan, dan merupakan standar industri. Dapat diaplikasikan pada bermacam tipe sistem, domain, dan metode atau proses [4].

1. Sebagai bahasa pemodelan yang general-purpose, difokuskan pada pokok himpunan konsep yang dapat dipakai bersama, dan menggunakan pengetahuan bersama dengan mekanisme perluasan.
2. Sebagai bahasa pemodelan yang mudah diaplikasikan, dapat diaplikasikan untuk bermacam tipe sistem (software dan non- software), domain (bisnis vs. software), dan metode atau proses.
3. Sebagai bahasa pemodelan standart industri, bukan merupakan bahasa yang tertutup atau satu-satunya, tapi bersifat terbuka dan sepenuhnya dapat diperluas.

UML berfungsi memungkinkan untuk menangkap, mengkomunikasikan, mengembangkan strategi, taktik dan mengoperasikan pengetahuan untuk meningkatkan nilai tambah melalui peningkatan kualitas, mereduksi biaya, mereduksi time-to-market.

2.5.2 Diagram UML

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[4]:

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.
2. *Use case scenario*

Sebuah artikulasi verbal pengecualian terhadap perilaku utama yang dijelaskan oleh kasus penggunaan utama. Use case skenario merupakan hasil instansiasi dari setiap use case. Terbagi menjadi tiga bagian, yaitu:

a. Identifikasi dan inisiasi.
b. Step perfomed.
c. Kondisi, asumsi dan pertanyaan.

3. *Activity* diagram

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

4. *Sequence diagrams*

   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).

5. *Class diagrams*

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

2.6 Adobe Flash

Flash merupakan program atau software yang ditujukan untuk membuat animasi, desain, web, aplikasi, *game*, dan sebagaianya. Flash lebih dikenal khalayak ramai sebagai software pembuat animasi. Seiring perkembangan zaman yang semakin maju dan pesat dalam bidang teknologi khususnya dalam bidang teknologi informasi, flash sampai sekarang telah mencapai versi 11 atau lebih dikenal dengan Adobe Flash CS5 Profesional yang dirilis oleh perusahaan software ternama di Amerika Serikat, yaitu Adobe System Incorporated.

Flash CS5 yang sekarang banyak dikenal dengan kemampuan yang andal dalam membuat animasi-animasi, tetapi tidak jarang yang mengenal sisi lain dari kelebihan

Flash ini. Salah satunya adalah mempunyai bahasa pemrograman yang sangat ampuh dalam segala hal, yaitu Action Script yang sekarang sudah dirilis samapai ke versi 3.0 yang dimiliki oleh Flash CS3,CS4, dan CS5.

Adobe Flash CS5 menawarkan beberapa pembaruan yang komplet dengan penawaran penggunaan lebih menyenangkan. Fasilitas seperti 3D Effect atau transformations yang sangat menarik dapat digunakan untuk membuat efek-efek animasi 3 dimensi yang menarik. Berbagai fasilitas dan fitur terbaru telah disediakan untuk kemudahan dalam pengolahan para penggunanya. Adobe Flash CS5 merupakan penyempurnaan dari versi sebelumnya (CS4) [8].