BAB II

LANDASAN TEORI

2.1 Pertani Indonesia

Pertani di Indonesia sebagian besar mengelola pertanian tanaman padi, hal ini disebabkan bahan pokok untuk kebutuhan hidup di Indonesia untuk bahan konsumsi sehari-hari berasal dari tanaman padi. Namun dalam mengelola pertanian padi, para petani indonesia sering dijumpai berbagai masalah dalam proses pengelolaan tanaman padinya. Salah satu penghambat pengelolaan padi di Indonesia karena terjadinya serangan hama yang sering meruksak tanaman padi di ladang pertanian, adapun hama-hama yang sering menyerang pertanian di Indonesia antara lain sebagai berikut [6] :

A. Walang Sangit *(Leptocorixa acuta*)

Walang sangit merupakan hama yang menghisap cairan bulir pada fase masak susu. Kerusakan yang ditimbulkan walang sangit menyebabkan beras berubah warna, mengapur serta hampa. Hal ini dikarenakan walang sangit menghisap cairan dalam bulir padi. Fase tanaman padi yang rentan terserang hama walang sangit adalah saat tanaman padi mulai keluar malai sampai fase masak susu. Pengendalian dianjurkan dilakukan pada saat gabah masak susu pada umur 70-80 hari setelah tanam dengan disemprot *insektisida Greta 500EC* .

B. Ulat Grayak (*Armyworm*)

Hama ulat grayak menyerang tanaman dengan memakan daun dan hanya meninggalkan tulang daun dan batang. Larva ulat grayak menyerang tanaman padi sejak di persemaian sampai fase pengisian. Serangan akan parah saat musim kemarau dan tanaman kekurangan air. Untuk pengendaliannya dianjurkan menyemprot dengan *Matrix* 200EC dan *Promectin 18EC* .

C. Burung (*Lonchura spp*.)

Burung menyerang tanaman pada fase masak susu sampai padi dipanen. Burung akan memakan langsung bulir padi yang sedang menguning sehingga menyebabkan kehilangan hasil secara langsung. Selain itu burung juga mengakibatkan patahnya malai padi. . Untuk pengendaliannya dianjurkan menyemprot dengan pestisida nabati ekstrak buah serut 90 cc.

D. Tikus Sawah

Tikus merusak tanaman pada semua fase pertumbuhan dan dapat menyebabkan kerusakan besar apabila tikus menyerang pada saat primodia. Tikus akan memotong titik tumbuh atau memotong pangkal batang untuk memakan bulir gabah. Pengendalian dengan menggunakan rodentisida *Brodirat 0,005BB*.

2.2 Definisi *game*

Game adalah permainan yang biasanya melibatkan player berinteraksi dengan alat pengendali untuk menghasilkan umpan balik secara visual dalam sebuah layar video [4]. Perangkat input yang digunakan untuk memanipulasi permainan video disebut pengendali permainan, dan bervariasi di seluruh platform. Sebagai contoh controller mungkin hanya terdiri dari sebuah tombol dan *joystick*, sementara yang lain mungkin mempunyai selusin tombol dan satu atau lebih *joystick*. Video game biasanya menggunakan sarana tambahan menyediakan interaktivitas dan informasi untuk pemain. Audio hampir *universal*, menggunakan perangkat reproduksi suara, seperti *speaker* dan *headphone*. Umpan balik lain mungkin datang melalui *peripheral haptic*, seperti getaran atau kekuatan, dengan getaran kadang-kadang digunakan untuk mensimulasikan umpan balik kekuatan.

2.3 *Game 2D*

Game dua dimensi dapat diketahui berdasarkan ruangnya yang hanya memiliki dua sisi (X dan Y). Sedangkan untuk gambarnya sendiri dapat menggunakan vector maupun *Bitmap*.Untuk membuat animasi bergerak seperti berjalan, melompat, berlari, dan lainnya maka harus membuat gambar satu persatu yang disebut dengan *frame*. Kerealisasian gerakan ditentukan dari gambar yang dibuat, jumlah gambar (*frame*) yang digunakan, serta hitungan gambar per detik (*frame* per *second*) semakin tinggi hitungan gambar per detik maka semakin mulus gerakan yang akan dihasilkan. *Game 2D* atau dua dimensi adalah konsep di mana semua objek berada pada satu bidang datar. Pemain *game 2D* tidak dapat bergerak bebas ke segala sisi, gerakan pada *game 2D* dibatasi hanya horisontal dan vertikal atau secara koordinat gerakan dibatasi hanya dapat bergerak pada sumbu X danY. Berdasarkan pergerakannya, *game 2D* dapat dibagi menjadi :

1. Game *statis* adalah *game* yang gambar latar dan tempat *game 2D* tidak bergerak sama sekali contohnya adalah *tetris*.
2. Side *Scrolling* adalah *game* yang saat dimainkan, kamera akan bergeser ke kanan atau ke kiri dengan kecepatan sesuai dengan gerakan dan kecepatan karakter yang dimainkan pada *game* tersebut, contohnya seperti *super Mario bross*, *Sonic* dan *Megaman*.

Pada *side scrolling game* sendiri dapat juga ditemukan game dimana screen yang tidak bergerak mengikuti karakter *game*, ada juga jenis game yang harus melewati berbagai rintangan, menembak musuh, mengkoleksi berbagai item seiring dengan screen yang bergerak sebelum screen tersebut terlewati.

2.4 *Game Desktop*

Game Berbasis *Desktop* merupakan permainan video yang bisa berjalan secara *independen* atau berjalan sendiri dalam sistem operasi tertentu dengan cara menginstallasi (memasamg aplikasi) dalam sistem operasi komputer (*PC*) sehingga bisa berjalan tanpa membutuhkan jaringan *internet* [11]. *Game desktop*

ini mulai dikenal seiring munculnya sistem operasi windows, karena setiap periode sistem operasi windows yang diluncurkan di dalamnya terdapat permainan yang melengkapi hiburan berbentuk multimedia untuk para penggunanya. *Game* sendiri diambil dari bahasa Inggris yang diterjemahkan yang artinya permainan. Di dalam era perkembangan era perkembangan teknologi yang pesat ini, seperti halnya permainan kartu, catur dan lain-lainnya dapat ditemui melalui dunia *virtual* atau yang biasa dimainkan di dalam komputer. Dengan perkembangan teknologi sekarang ini munculah berbagai sarana permainan, misalnya Playstation maupun *PC* dari yang berbasis individu ataupun multiplayer [8].

2.5 *Genre Game*

Saat ini telah banyak ditemukan berbagai *game*, baik itu game *PC*, *game console* (seperti *PS, XBOX, PSP, Nintendo DS*), *game HP* (*mobile game*) dan *game flash*. Dari banyaknya *game* yang ada, berdasarkan *gameplaynya* (cara memainkannya), *game* dapat diklasifikasikan menjadi beberapa jenis. Dalam hal ini klasifikasi *game flash* bisa dilihat sebagai berikut [8]. :

1. *Board game*

Permainan ini dapat diistilahkan sebagai "permainan papan". Pada jenis *game* ini, pemain diberikan sebuah tampilan yang berisi tentang masalah untuk diselesaikan. Contoh dari game ini antara lain: *Pipe dream, Hangaroo, Rotation,* Catur, permainan kartu dan sebagainya .

2. *Arcade*

Permaian bertipe arcade merupakan *game* yang menguci kecepatan tangan dari pemainnya. Pada permainan bertipe *arcade*, semakin tinggi level permainan, permainan akan berjalan semain cepat. Contoh dari *game* bertipe *arcade* adalah : *zuma, feeding frenzy, Pacman, Arcanoid, Pong, Baba ball* dan sebagainya .

3. *Action*

Berbeda dengan *game* bertipe *arcade,* game bertipe action menjadikan pemain mengendalikan karakter utama dalam *game* tersebut untuk melakukan beberapa kegiatan (*action*) seperti melompat, menembak dan sebagainya. Contoh dari *game* bertipe action antara lain : *super mario, mocil, petualangan paddle pop, alloy tease exile* dan sebagainya .

4. *Shooting*

Permainan bertipe *shooting* sebagian besar menggunakan mouse sebagai alat pengendalinya. Pada *game* ini pemain seolah-olah berperan sebagai penembak *(first person)* atau pemain mengendalikan seorang penembak (*third person*). 4. *ShootingGame* bertipe *shooting* sebagian besar menggunakan *mouse* sebagai alat pengendalinya. Pada game ini pemain seolah-olah berperan sebagai penembak (*first person*) atau pemain mengendalikan seorang penembak *(third person*).Contoh game bertipe shooting antara lain : *Duck hunt, Counter Strike, Onimusha* dan sebagainya .

5. *Fighting*

Permainan bertipe *fighting* pada dasarnya sama dengan game bertipe *action*, hanya saja *game* bertipe fighting pemain mengendalikan sebuah karakter untuk berkelahi dengan karakter lain sampai salah satu karakter kalah. Contoh dari game bertipe fighting antara lain : *street fighter, tekken, duel*, pencak silat dan sebagainya .

6. *Racing*

Permaian betipe racing pada dasarnya adalah sebuah permainan menggerakkan kamera. Pemain diberikan sebuah kendaraan atau sejenisnya untuk menempuh rute tertentu. Contoh dari game bertipe racing antara lain : NFS, *Auto bahn, MotoGP, Formula 1* dan sebagainya .

7. *Simulation*

Permainan bertipe simulasi adalah sebuah *game* yang mensimulasikan suatu kegiatan yang sesungguhnya. Contoh dari *game* bertipe simulasi antara lain : *tycoon, simulator* pesawat, *burger empire* dan sebagainya .

8. *Real Time Strategi*

Permainan bertipe RTS memposisikan pemain sebagai seorang pemimpin yang mengatur sesuatu (bisa berupa pasukan, koloni, kerajaan dan sebagainya). Contoh dari *game* bertipe RTS antara lain: *war commander, empire earth, romance of the three kingdom* dan sebagainya .

9. *Role Playing Game*

Permainan bertipe RPG pemain memerankan sebuah karakter dalam *game.* Berbeda dengan *game* bertipe *action*, pada game RPG hal yang diutamakan adalah cerita dalam *game*. Selain itu didalam *game* bertipe RPG biasanya terdapat sub *game* dengan tipe lain (seperti bertipe *fighting, action* atau RTS). Contoh *game* bertipe RPG antara lain : *BOBO games*, *Zelda*, *Megaman*, *Final Fantasy* dan sebagainya .

10.*MMO*

MMO (*Masive Multiplayer Online*), merupakan *game* yang dapat dimainkan secara bersama-sama pada *internet browser*. Hal yang diutamakan dalam sebuah *game* bertipe MMO adalah kebersamaan dengan pemain lain. Contoh *game* bertipe MMO antara lain : *ragnarok, pangya, gun bound, seal* dan sebagainya. *Game flash* bertipe MMO masih sangat sedikit-hal tersebut dikarenakan untuk membuat *game flash* MMO diperlukan software lain (*server side software*) seperti *PHP, CGI, Fire Fox*, dan sebagainya. Contoh dari *game flash* bertipe MMO antara lain : *Dofus* dan *battle-on* .

2.6 Kecerdasan Buatan

Kecerdasan buatan (*Artificial Intelligence*) didefinisikan sebagai kecerdasan yang ditunjukkan oleh suatu *entitas* buatan. Kecerdasan diciptakan dan dimasukkan ke dalam suatu mesin (komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia [9]. Beberapa macam bidang yang menggunakan kecerdasan buatan salah satunya permainan komputer **(***games***).**

Perkembangan game saat ini tidak lepas dari kecerdasan buatan (*artificiall intelligence*). Berdasarkan perkembangan *game* yang pesat pada masa ini, maka tidak dipungkiri bahwa dibutuhkan sesuatu yang berbeda pada rule permainannya. Hal ini sangat berkaitan dengan kecerdasan buatan (artificiall intelligence) yang diterapkan pada game. Sebelumnya, sebuah sistem *game*, jika sudah dimainkan sampai tuntas oleh seorang , maka ketika *player* yang sama memulai lagi permainan dari awal, maka *rule* permainannya akan sama. Namun untuk saat ini sesuai dengan perkembangan *game* dan kecerdasan buatan yang diterapkan. sistem dalam *game* sudah dapat belajar mengenali pola permainan dari player dan ketika player tersebut memulai permainan kembali, maka sistem ini akan menggunakan *rule* yang berbeda untuk pemain yang sama ini, sehingga game menjadi lebih menarik dan menantang untuk dimainkan.

2.6.1 Algoritma *fuzzy logic*

Pada game ini *fuzzy logic* digunakan untuk menentukan variasi prilaku yang dilakukan oleh NPC. Dengan adanya fuzzy logic masing-masing NPC dapat merubah perilaku berbasis perubahan variabel masukan. Metode fuzzy yang digunakan adalah metode mamdani karena metode ini menghasilkan keluaran yang bervariasi, sehingga dapat mewakili nilai perilaku yang telah dirancang. Logika fuzzy sendiri merupakan suatu cara yang tepat untuk memetakan suatu ruang input ke dalam suatu ruang output. Logika *Fuzzy* diperkenalkan oleh seorang profesor “Lutfi Astor Zadeh” pada tahun 1962
konsep himpunan fuzzy memiliki 2 atribut, yaitu :

1. Linguistik, yaitu nama suatu kelompok yang mewakili suatu keadaan tertentu dengan menggunakan bahasa alami, misalnya dekat, sedang, jauh mewakili variable jarak.
2. Numeris, yaitu suatu nilai yang menunjukan ukuran dari suatu variabel, misalnya 10, 35, 40 dan sebagainya.

. Dalam sistem kontrol logika *fuzzy* terdapat beberapa tahapan operasional yang meliputi [10]:

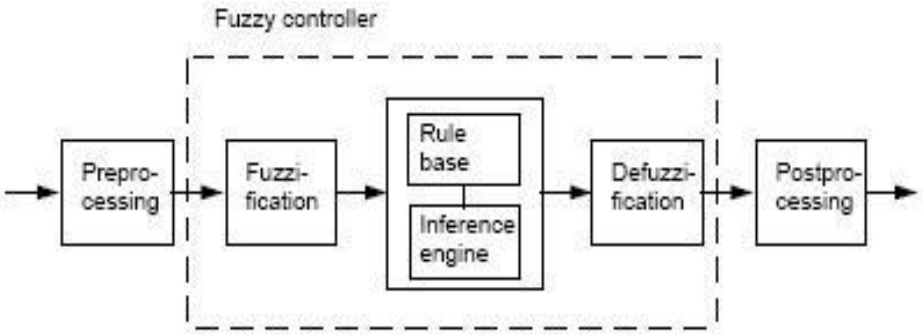
1. *Fuzzyfikasi*

   Merupakan proses dari nilai dua parameter yang diperoleh dari hasil ekstraksi yang selanjutnya dikonversi ke bentuk masukan *fuzzy*, nilai ini dinamakan variabel *linguistik*
2. Penalaran (*Inference Machine*)

   Merupkan Proses untuk menentukan penalaran (keputusan yang logis) dalam pengambilan keputusan yang di terapkan pada Sistem
3. Komposisi aturan (*Rule Based*)

   merupakan proses untuk menentukan aturan *fuzzy* yang akan digunakan dalam sistem.
4. *Defuzzyfikasi*

   Defuzzifikasi merupakan proses terakhir dalam suatu sistem logika *fuzzy* dimana tujuannya adalah mengkonversi setiap hasil dari *inference engine* yang diekspresikan dalam bentuk *fuzzy* set ke suatu bilangan real.



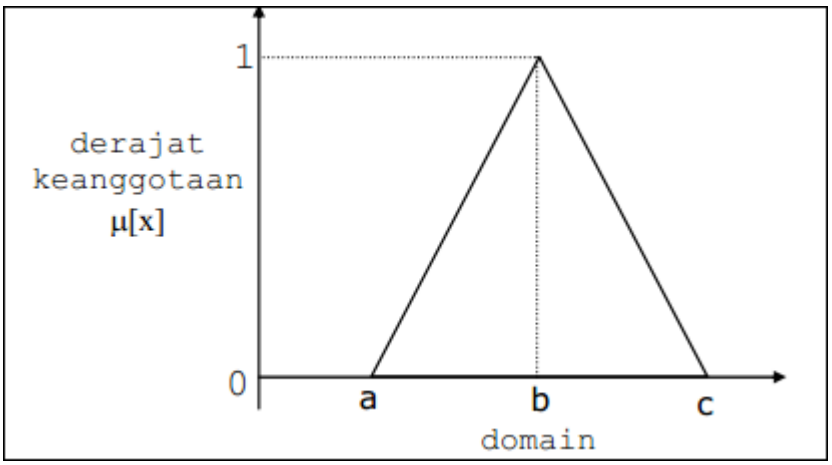
Gambar 2.1 Proses *Fuzzy controller*

A. Fungsi keanggotaan

Fungsi Keanggotaan (membership function) adalah suatu kurva yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaannya (sering juga disebut dengan derajat keanggotaan) yang memiliki interval antara 0 sampai 1. Salah satu cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi. Ada beberapa fungsi yang bisa digunakan :

1. Kurva Segitiga

Kurva Segitiga pada dasarnya merupakan gabungan antara 2 garis (linear) seperti terlihat pada Gambar



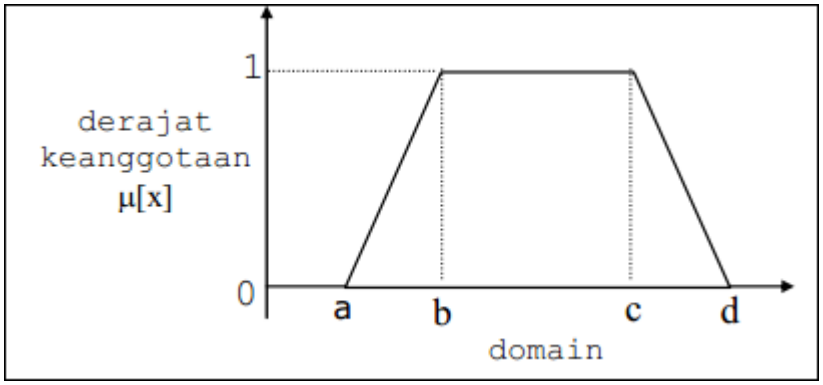
Gambar 2.2 Contoh Kurva segitiga

Fungsi Keanggotaan :

$$\mu[x] = \begin{cases} 0; & x \le a \text{ atau } x \ge c \\ (x - a)/(b - a); & a \le x \le b \\ (b - x)/(c - b); & b \le x \le c \end{cases} \quad (2.1)$$

2. Kurva Trapesium

Kurva Segitiga pada dasarnya seperti bentuk segitiga, hanya saja ada beberapa titik yang memiliki nilai keanggotaan



Gambar 2.3 Contoh Kurva Trapesium

Fungsi Keanggotaan :

$$\mu[x] = \begin{cases} 0; & x \le a \text{ atau } x \ge d \\ (x - a)/(b - a); & a \le x \le b \\ 1; & b \le x \le c \\ (d - x)/(d - c); & x \ge d \end{cases} \quad (2.2)$$

B. Metode Mamdani

Metode Mamdani sering juga dikenal dengan nama metode Max-Min. Metode ini pertama kali Ebrahim Mamdani pada tahun 1975 [11]. Berikut adalah tahapan-tahapan dari proses metode Mamdani :

1. Pembentukan Himpunan Fuzzy

   Merupakan proses menentukan variabel fuzzy dan himpunan fuzzynya, kemudian menentukan derajat keanggotaan antara data *input* dengan himpunan fuzzy yang telah didefinisikan untuk setiap variabel *input*.

2. Aplikasi Fungsi Implikasi

   Fungsi implikasi yang digunakan adalah *min*. melakukan implikasi fuzzy berdasar pada kuat penyulutan dan himpunan fuzzy terdefinisi untuk setiap variabel keluaran di dalam bagian konsekuensi dari setiap aturan [2]. Hasil implikasi fuzzy dari setiap aturan ini kemudian digabungkan untuk menghasilkan keluaran infrensi fuzzy.

3. Komposisi Aturan

   Merupakan inferensi yang diperoleh dari korelasi beberapa aturan.

4. Defuzzykasi

   Merupakan himpunan fuzzy dari komposisi aturan-aturan fuzzy, sedangkan *output* yang dihasilkan merupakan suatu bilangan pada domain himpunan *fuzzy* tersebut.

Pada proses defuzzykasi ada beberapa metode yang digunakan salah satunya menggunakan metode centroid. Dimana pada metode centroid, solusi *crisp output* diperoleh dengan cara mengambil titik pusat daerah *fuzzy* [13]. Berikut adalah rumus perhitungan dari metode centroid :

$$z = \frac{\sum_{j=1}^{n} z_j \mu(z_j)}{\sum_{j=1}^{n} \mu(z_j)} \tag{2.3}$$

Keterangan :

Z : Nilai yang dicari untuk memperoleh nilai defuzzikasi.

n : Banyaknya nilai tiap variabel.

j : Banyaknya perhitungan pada variabel Z.

Zj : Hasil yang diperoleh dari perhitungan variabel Z.

μ : Nilai yang diperoleh dari pembentukan himpunan fuzzy.

2.7 Aplikasi Pembangun Perangkat Lunak

Aplikasi yang digunakan untuk membangun aplikasi ini adalah *OOP*, *UML*, *Adobe Flash* , *Action Script* dan *Adobe Photoshop*. Berikut ini penjelasannya **:**

2.7.1 *OOP* (*Object Oriented Programming*)

Pemrograman berorientasi objek bekerja dengan baik ketika di barengi dengan *Object-oriented analysis* and Design Process [14]. Sebab *OOP* merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Bandingkan dengan logika pemrograman terstruktur. Setiap

objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya. Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung *OOP* mengklaim bahwa *OOP* lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan *OOP* lebih mudah dikembangkan dan dirawat. Konsep dasar dari Pemrograman Berorientasi Objek Pemrograman orientasi-objek menekankan konsep berikut [14] :

1. Kelas (*Class*) adalah penggambaran satu objek yang memiliki atribut dan *behavior* yang sama. Kelas mirip tipe data pada pemrograman non objek, tapi lebih komprehensif karena terdapat struktur sekaligus karakteristiknya.kita dapat membentuk kelas baru yang lebih spesifik dari kelas generalnya.kelas dan objek merupakan jantung dari pemrograman berorientasi objek untuk menghasilkan program jenis ini sangat penting untuk selalu berfikir dalam konsep objek.

2. Objek merupakan teknik dalam penyelesaian masalah yang kerap muncul dalam pembangunan perangkat lunak. Selain itu Objek membungkus data dan fungsi bersama menjadi suatu unit dalam sebuah program komputer, objek merupakan dasar dari modularitas dan struktur dalam sebuah program komputer berorientasi objek.

3. *Abstraksi* Kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti. Setiap objek dalam sistem melayani sebagai model dari "pelaku" abstrak yang dapat melakukan kerja, laporan dan perubahan keadaannya, dan berkomunikasi dengan objek lainnya dalam sistem, tanpa mengungkapkan bagaimana kelebihan ini diterapkan. Proses, fungsi atau metode dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan.

4. *Enkapsulasi* Memastikan pengguna sebuah objek tidak dapat mengganti keadaan dalam dari sebuah objek dengan cara yang tidak layak, hanya metode dalam objek tersebut yang diberi izin untuk mengakses keadaannya. Setiap objek mengakses interface yang menyebutkan bagaimana objek lainnya dapat berinteraksi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada *representasi* dalam objek tersebut.

5. *Polimorfisme* melalui pengiriman pesan. Tidak bergantung kepada pemanggilan subrutin, bahasa orientasi objek dapat mengirim pesan, metode tertentu yang berhubungan dengan sebuah pengiriman pesan 20 tergantung kepada objek tertentu di mana pesa tersebut dikirim. Contohnya, bila sebuah burung menerima pesan "gerak cepat", dia akan menggerakan sayapnya dan terbang. Bila seekor singa menerima pesan yang sama, dia akan menggerakkan kakinya dan berlari. Keduanya menjawab sebuah pesan yang sama, namun yang sesuai dengan kemampuan hewan tersebut. Ini disebut polimorfisme karena sebuah variabel tungal dalam program dapat memegang berbagai jenis objek yang berbeda selagi program berjalan, dan teks program yang sama dapat memanggil beberapa metode yang berbeda di saat yang berbeda dalam pemanggilan yang sama. Hal ini berlawanan dengan bahasa fungsional yang mencapai polimorfisme melalui
penggunaan fungsi kelas-pertama. Dengan menggunakan *OOP* maka dalam melakukan pemecahan suatu masalah kita tidak melihat bagaimana cara menyelesaikan suatu masalah tersebut (terstruktur) tetapi objek-objek apa yang dapat melakukan pemecahan masalah
tersebut. Sebagai contoh anggap kita memiliki sebuah departemen yang memiliki manager, sekretaris, petugas administrasi data dan lainnya. Misal manager tersebut ingin memperoleh data dari bag administrasi maka manager tersebut tidak harus mengambilnya langsung tetapi dapat menyuruh petugas bag administrasi untuk mengambilnya. Pada kasus tersebut seorang manager tidak harus mengetahui bagaimana cara mengambil data tersebut tetapi manager bisa mendapatkan data tersebut melalui objek petugas adminiistrasi. Jadi

untuk menyelesaikan suatu masalah dengan kolaborasi antar objek-objek yang ada karena setiap objek memiliki deskripsi tugasnya sendiri.

2.7.2 *UML* (*Unified Modeling Language*)

UML (Unified Modeling Language) adalah bahasa spesifikasi standar untuk mendokumentasikan, menspesifikasikan, dan membangun system (Flowler, 2006). Unified Modeling Language (*UML*) adalah himpunan struktur dan teknik untuk pemodelan desain program berorientasi objek (*OOP*) serta aplikasinya. *UML* adalah metodologi untuk mengembangkan sistem *OOP* dan sekelompok perangkat tool untuk mendukung pengembangan sistem tersebut. UML mulai diperkenalkan oleh *Object* Management Group, sebuah organisasi yang telah mengembangkan model, teknologi, dan standar *OOP* sejak tahun 1980-an. Sekarang UML sudah mulai banyak digunakan oleh para praktisi *OOP* [11]. Logo *UML* dapat dilihat pada gambar 2.2.



Gambar 2.4 Logo *UML*

UML suatu bahasa yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan suatu sistem informasi. *UML* dikembangkan sebagai suatu alat untuk analisis dan desain berorientasi objek oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson. Namun demikian *UML* dapat digunakan untuk memahami dan mendokumentasikan setiap sistem informasi. Penggunaan *UML* dalam industri terus meningkat, ini merupakan standar terbuka yang menjadikannya sebagai bahasa pemodelan yang umum dalam industri peranti lunak dan pengembangan sistem. Bagian-

bagian utama dari *UML* adalah view, diagram, model element, dan general mechanism [11]. Diagram berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu 23 dari sistem. Sebuah diagram merupakan bagian dari suatu *view* tertentu dan ketika digambarkan biasanya

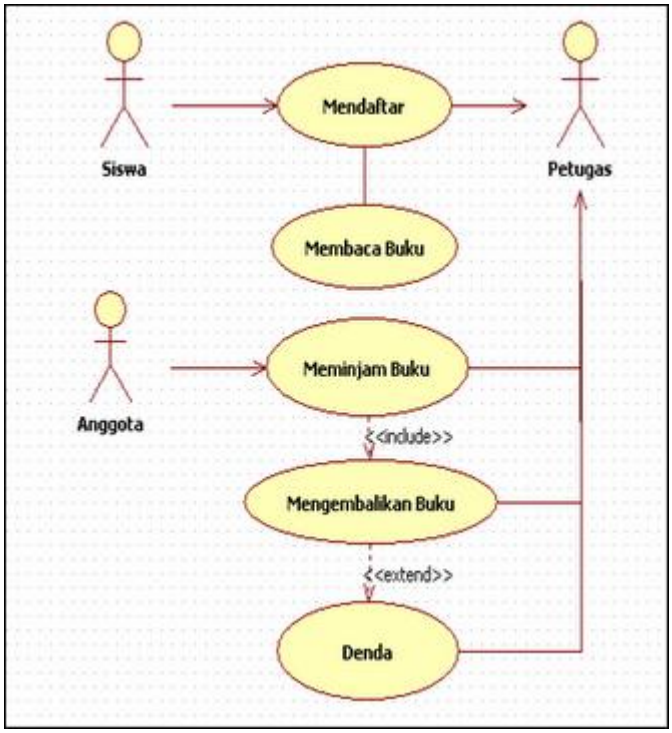
2.7.2.1 *Usecase* Diagram

Usecase [14]. adalah abstraksi dari interaksi antara system dan actor. *Usecase* bekerja dengan cara mendeskripsikan tipe interaksi antara user sebuah system dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah system dipakai. *Usecase* merupakan konstruksi untuk mendeskripsikan bagaimana system akan terlihat di mata user. Sedangkan *Usecase* diagram memfasilitasi komunikasi diantara analis dan pengguna serta antara analis dan client.

Contoh pada *use case diagram* perpustakaa mempunyai Lima fungsionalitas yang terdiri dari :

1. mendaftar artinya siswa harus melakukan pendaftaran sebelum membaca buku di perpustakaan baik menjadi anggota ataupun hanya pengunjung saja.
2. membaca buku artinya siswa yang telah mendaftarkan diri ke petugas perpustakaan boleh membaca buku dalam perpustakaan
3. meminjam buku artinya jika siswa yang mendaftar ke petugas menjadi anggota perpustakaan maka diperbolehkan meminjam buku perpustakaan.
4. mengembalikan buku artinya anggota yang telah meminjam buku harus mengembalikan lagi ke petugas perpustakaan jika masa pinjaman telah habis.
5. denda artinya anggota yang mengembalikan buku melebihi masa peminjaman maka akan dikenakan denda.

Usecase pada Gambar 2.3 merupakan contoh dari *use case* diagram sistem perpustakaan



Gambar 2.5 Contoh *Usecase* diagram sistem perpustakaan

2.7.2.2 *Class* Diagram

Class adalah dekripsi kelompok obyek-obyek dengan properti, perilaku (operasi) dan relasi yang sama. Sehingga dengan adanya class diagram dapat memberikan pandangan global atas sebuah system. Hal tersebut tercermin dari *class- class* yang ada dan relasinya satu dengan yang lainnya. Sebuah sistem biasanya mempunyai beberapa class diagram. *Class* diagram sangat membantu alam visualisasi struktur kelas dari suatu system.

Class memiliki tiga area pokok yaitu Nama Kelas, Atribut dan Metode. Atribut dan metode dapat memiliki salah satu sifat berikut :

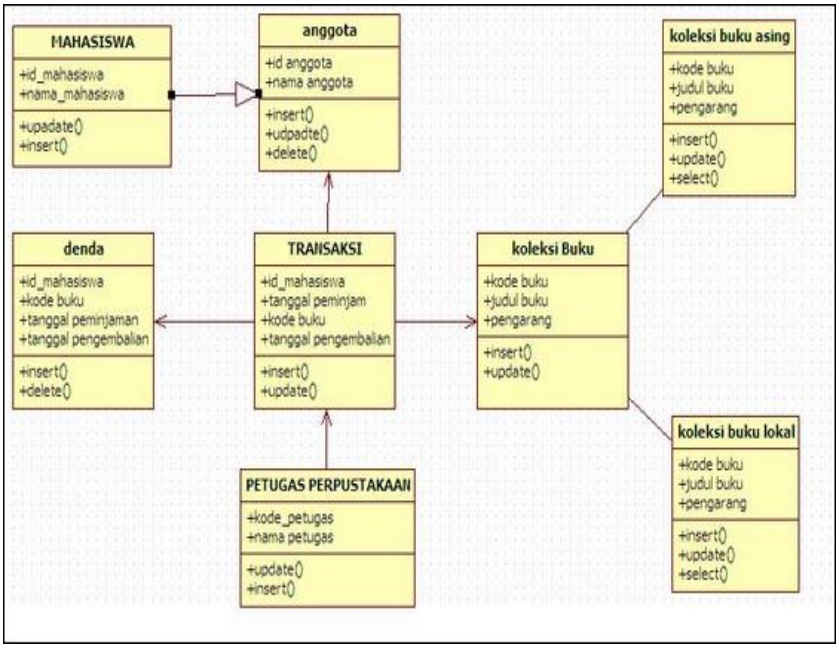
1. *Private* artinya data yang dikendalikan oleh suatu kelas tertentu dan tidak dapat diakses oleh objek-objek yang berbeda kelas. Satu ciri metode *sharing* yang bersifat *private* adalah hanya fungsi-fungsi atau metode – metode dalam kelas/objek yang bersangkutan yang dapat mengakses metode atau variabel serta atribut dalam kelas tanpa batasan. Biasanya tanda (-) digunakan untuk atribut atau metode yang bersifat *private*.

2. *Protected* artinya atribut atau metode yang bersifat *protected* hanya bisa diakses oleh kelas yang bersangkutan serta kelas-kelas yang merupakan turunannya mengikuti hierarki dari generalisasi. Biasanya tanda (#) digunakan untuk atribut atau metode yang bersifat *protected*.
3. *Public* artinya metode atau atribut yang bersifat *public* dapat diakses oleh semua objek dalam aplikasi tertentu tanpa batasan apa pun. Biasanya tanda (+) digunakan untuk atribut atau metode yang bersifat *public*.

Contoh *class* diagram pada sistem informasi perpustakaan memiliki delapan class antara lain:

1. class mahasiswa memiliki atribut : id_mahasiswa dan nama_mahasiswa dengan method : update dan insert dan bersifat public.
2. class anggota memiliki atribut : id anggota dan nama anggota dengan method :insert, update , delete dan bersifat public.
3. class koleksi buku memiliki atribut : kode buku, judul buku dan pengarang dengan method : insert, update,select. Dan bersifat public.
4. class denda memiliki atribut : id mahasiswa, kode buku, tgl pinjam, tgl kembali dengan method : insert ,update dan bersifat public.
5. class transaksi memiliki atribut : atribut : id mahasiswa, kode buku, tgl pinjam, tgl kembali dengan method : insert ,update dan bersifat public.
6. class koleksi buku memiliki atribut : atribut : id mahasiswa, kode buku, tgl pinjam, tgl kembali dengan method : insert ,update dan bersifat public.
7. class petugas perpustakaan memiliki atribut : atribut : id mahasiswa, kode buku, tgl pinjam, tgl kembali dengan method : insert ,update dan bersifat public.
8. class koleksi buku loka memiliki atribut : atribut : id mahasiswa, kode buku, tgl pinjam, tgl kembali dengan method : insert ,update dan bersifat public.

Class diagram pada Gambar 2.4 merupakan contoh class diagram pada sistem informasi perpustakaan.



Gambar 2.6 Contoh *Class* Diagram pada sistem Peminjaman buku perpustakaan.

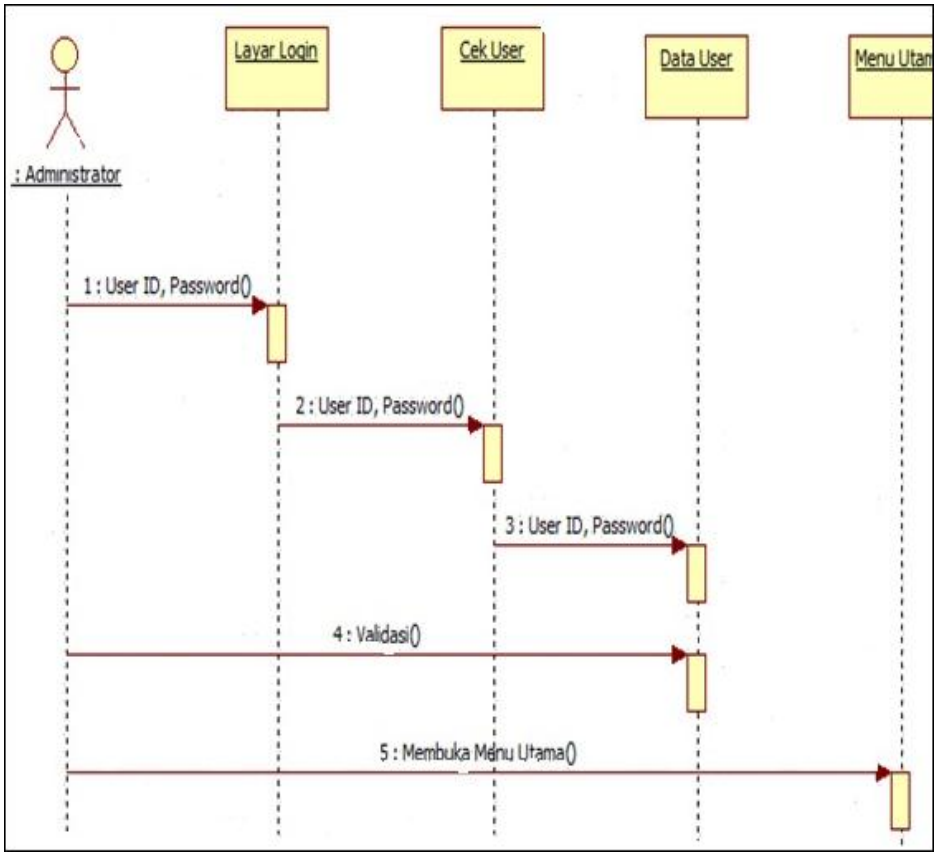
2.7.2.3 *Sequence* Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah scenario. Kegunaannya untuk menunjukkan rangkaian pesan yang dikirim antara object juga interaksi antara object, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.

Contoh pada *sequence diagram login* mempunyai interaksi sebagai berikut

a. Administrator (Aktor) melakukan *login*, maka *method User ID*, *Password* () memanggil kelas Layar *Login*.
b. Kelas Layar *Login* memanggil kelas Cek *User* untuk mengecek *User ID* dan *Password*.
c. Kelas Cek *User* memanggil kelas Data *User* untuk memvalidasi *User ID* dan *Password* di *database*.
d. Apabila *User ID* dan *Password* valid maka Administrator beerhasil melakukan *Login*.

Sequence Diagram pada Gambar 2.5 merupakan contoh dari *sequence* diagram login.



Gambar 2.7 Sequence Diagram login.

2.7.2.4 *Activity* Diagram

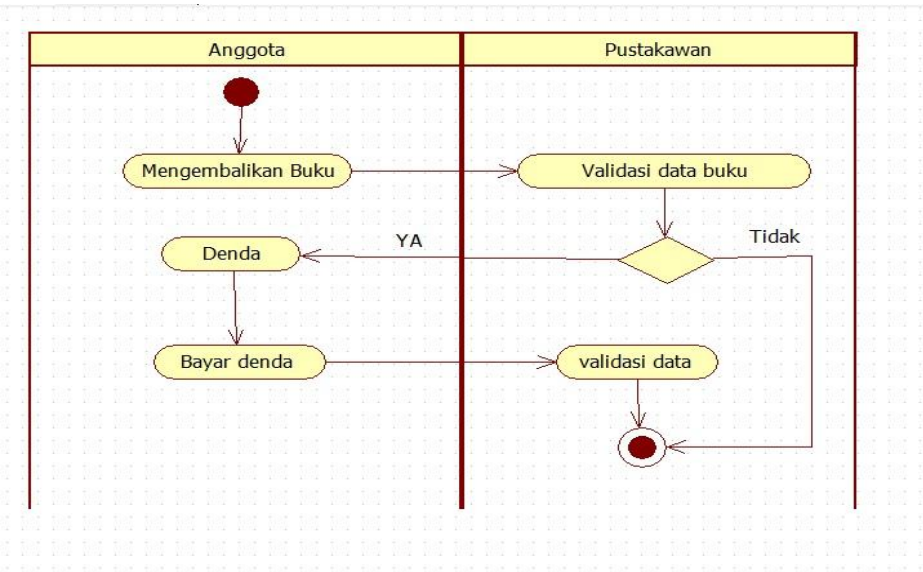
Activity Diagram menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *Usecase* atau interaksi,

Sebuah aktifitas dapat direalisasikan oleh suatu *use case* atau lebih. Aktifitas menggambarkan proses yang berjalan sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktifitas. Contoh pada *activity diagram* pengembalian buku perpustakaan, pada *activity diagram* pengembalian buku perpustakaan mempunyai alir aktifitas sebagai berikut :

a. Anggota perpustakaan mengembalikan buku perpustakaan kepada pustakawan.

b. Pustakawan memvalidasi data buku yang dipinjam oleh anggota perpustakaan.
c. Apabila anggota perpustakaan mengembalikan buku melebihi batas waktu maka anggota perpustakaan akan mendapat denda dengan membayar sejumlah uang.
d. Kemudian pustakawan memvalidasi data anggota perpustakaan.
e. Apabila anggota perpustakaan mengembalikan buku tidak melebihi batas waktu pengembalian maka anggota tidak mendapat denda.

contoh dapat dilihat pada gambar 2.6 merupakan *activity* diagram penegembalian buku perpustakaan.



Gambar 2.8 *Activity* Diagram pengembalian buku perpustakaan.

2.7.3 *Adobe Flash*

Adobe Flash (dahulu bernama *Macromedia Flash*) adalah salah satu perangkat lunak komputer yang merupakan produk unggulan *Adobe Systems*. *Adobe Flash* digunakan untuk membuat gambar vektor maupun animasi gambar tersebut. Berkas yang dihasilkan dari perangkat lunak ini mempunyai file *extension .swf* dan dapat diputar di penjelajah web yang telah dipasangi *Adobe Flash Player*. *Flash* menggunakan bahasa pemrograman bernama *ActionScript* yang muncul pertama kalinya pada *Flash* 5 [17].

2.7.4 *ActionScript*

ActionScript [17]. adalah Bahasa yang dikenali dan digunakan dalam *Flash* untuk melakukan berbagai hal dalam olah *movie*. bahasa pemrograman yang dibuat berdasarkan *ECMAScript* ini digunakan dalam pengembangan situs web dan perangkat lunak menggunakan platform *Adobe Flash Player*. *ActionScript* juga dipakai pada beberapa aplikasi basis data, seperti *Alpha Five*. Bahasa ini awalnya dikembangkan oleh *Macromedia*, tapi kini sudah dimiliki dan dilanjutkan perkembangannya oleh *Adobe*, yang membeli *Macromedia* pada tahun 2005. *ActionScript* terbaru saat ini adalah *ActionScript* 3.0. *ActionScript* 3.0 adalah bahasa terbaru dari edisi yang sebelumnya dikenal dengan *ActionScript* 2.0. *ActionScript* 3.0 memiliki beberapa kelebihan dibanding pendahulunya, antara lain fitur yang ditawarkan adalah file pada *ActionScript* 3.0 dapat dibuat terpisah saat runtime [14].

2.7.5 *Adobe Photoshop*

Adobe Photoshop adalah *Software* pengolahan foto yang sangat populer dan telah banyak para pengguna program ini yang mengakui kehebatan dari *Adobe photoshop* [16]. Selain itu perangkat lunak editor citra buatan *Adobe Systems* ini dikhususkan untuk pengeditan foto/gambar dan pembuatan efek. Perangkat lunak ini banyak digunakan oleh fotografer digital dan perusahaan iklan sehingga dianggap sebagai pemimpin pasar (*market leader*) untuk perangkat lunak pengolah gambar (foto) dan bersama *Adobe Acrobat* dianggap sebagai produk terbaik yang pernah diproduksi oleh *Adobe Systems*. Versi kedelapan aplikasi ini disebut dengan nama *Photoshop CS* (*Creative Suite*), versi sembilan disebut *Adobe Photoshop CS2*, versi sepuluh disebut *Adobe Photoshop CS3* , versi kesebelas adalah *Adobe Photoshop CS4* , versi keduabelas adalah Adobe *Photoshop CS5* , dan versi yang terakhir (ketiga belas) adalah *Adobe Photoshop CS6*.