BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Berikut ini beberapa penelitian yang berhubungan dengan penulisan skripsi adalah sebagai berikut :

Palipus Tarigan (2018), Sistem Pakar Untuk Mendiagnosa Penyakit Disentri Dengan Menggunakan Metode Hybrid Case Based. Dari hasil penelitian, pembahasan mendapatkan dan menghasilkan aplikasi sistem pakar, Setelah dilakukan pengujian dengan menggunakan beberapa data uji, aplikasi sistem pakar yang dirancang dapat mendeteksi penyakit Disentri. Dapat menentukan gejala-gejala yang berkaitan dengan penyakit disentri yaitu: demam, mual, diare, muntah, kram perut, tinja/feses berlendir, demam dan menggigil, penurunan berat badan, tinja/feses berdarah dan bernanah, dehidrasi, nyeri perut, sakit saat buang airbesar, lemas yang berlebihan, pendarahan pada rectum, tinja yang keluar sedikit. dari input data gejala yang dimasukkan dan memberikan hasil sesuai dengan jawaban pakar, Metode *hybrid case based* digunakan untuk melakukan penelusuran untuk mendapatkan hasil penyakit Disentri. Dengan demikian hasil dari penelurusan metode *hybrid case based* di dapat 78% hasil dari pengujian ini dinyatakan dengan kemungkina besar terkena disentri berdasarkan gejala-gejala yang di input *user*.

Chintami Febri Idris (2019), "Sistem Pakar Mendiagnosa Penyakit Polio Menerapkan Metode *Hybrid Case Based*". Berdasarkan hasil pengujian, dapat disimpulkan bahwa . Gejala-gejala pada penyakit polio yaitu ada yang ringan dan ada juga yang dapat mengakibatkan kelumpuhan pada anggota gerak tubuh. Akan tetapi secara umum gejala pada penyakit ini yaitu terjadinya demam pada penderita sekitar 2-5 hari yang kemudian disusul dengan lumpuhnya salah satu anggota gerak pada tubuh. Dari hasil penelitian, pembahasan mendapatkan dan menghasilkan aplikasi program sistem pakar diagnosis nilai tertinggi terdapat pada kasus 2 dengan nilai mencapai 100%.. Hasil kesimpulan dari aplikasi yang dibangun keakuratannya mencapai 100% dengan membandingkan, dengan adanya akses online berbasis web maka masyarakat dapat mendiagnosa kemungkinan penyakit polio yang dideritanya sebelum mengambil tindakan lebih lanjut seperti konsultasi ke dokter atau tes laboratorium di rumah sakit. Aplikasi sistem pakar ini dapat menjadi sarana untuk menyimpan pengetahuan tentang penyakit polio dari para pakar atau ahlinya.

Irpan, Muhammad Syahrizal, Imam Saputra (2018), "Sistem Pakar Untuk Mendiagnosa Kerusakan Mesin Bubut Menggunakan Metode Hybrid Case Based". Berdasarkan data kerusakan dan data gejala maka dikelompokkanlah data gejala tersebut menjadi basis kasus baik yang digunakan dalam pendekatan berdasarkan *case based* maupun berdasarkan *rule based* yang kemudian dijadikan sebagai paramter untuk menemukan solusi diagnose kerusakan mesin bubut. 1. Basis Data Kasus (*Case Based*) Penetapan basis kasus ini, dilakukan

berdasarkan pengalaman teknisi/pakar atau berdasarkan gejalagejala yang sering menimbulkan kerusakan mesin bubut yang sering terjadi.

2. Basis Data Aturan (*Rule Based*) Basis data aturan dibentuk berdasarkan data kerusakan dan gejala yang telah diuraikan pada tabel sebelumnya. Pembentukan data aturan ini merupakan pengelompokkan gejala-gejala yang menyebabkan setiap kerusakan mesin bubut.

Suci Fidyaningsih, Fahrul Agus, Septya Maharani (2016), "Sistem Pakar Diagnosa Penyakit Kucing Menggunakan Metode *Case-Based Reasoning".* Dari perancangan dan implementasi yang telah dilakukan, ada beberapa kesimpulan yang didapat dari hasil penelitian, antara lain:

1. Telah dibangun aplikasi Sistem Pakar Diagnosa Penyakit Kucing menggunakan metode *Case Based Reasoning*.

2. Metode *Case-Based Reasoning* dapat diimplementasikan pada aplikasi system pakar diagnosa penyakit kucing dengan tingkat akurasi sebesar 90%.

3. Menghasilkan output hasil diagnose penyakit kucing beserta solusi dan pencegahan berdasarkan 5 penyakit kucing yaitu *Scabies*, *Otitis*, Cacingan, *Ringworm*, dan *Rabies*.

M.Abdurrachman Irfandi, Ade Romadhony, Siti Saadah (2015), "Implementasi Sistem Pakar Diagnosa Penyakit Gigi DAN Mulut Menggunakan Metode *Hybrid Case-Based* Dan *Rule-Based Reasoning".* Berdasarkan pengujian dan analisis yang telah dilakukan, maka dapat disimpulkan

bahwa *Rule Based Reasoning* menggunakan ID3 mempunyai akurasi lebih baik dibandingkan *Rule Based Reasoning* menggunakan hasil wawancara. Hal ini dikarenakan hasil wawancara mempunyai *rule* dengan gejala yang lebih detail dibandingkan menggunakan ID3 yang menyederhanakan pohon keputusan. Kemudian system pakar dengan menggunakan *metode Hybrid Rule Based-Case Based Reasoning* memiliki akurasi lebih tinggi dibandingkan system pakar dengan metode *Rule Based Reasoning* saja.

II.2. Landasan Teori

Landasan teori sangat penting dalam sebuah penelitian terutama dalam penulisan skripsi, peneliti tidak bisa mengembangkan masalah yang mungkin di temui di tempat penelitian jika tidak memiliki acuan landasan teori yang mendukungnya.

II.2.1. Konsep Dasar Sistem

Pengertian sistem adalah seperangkat komponen yang saling berhubungan dan saling berkerjasama untuk mencapai beberapa tujuan. Sistem adalah sekelompok unsur yang erat hubungannya satu dengan yang lainnya, yang berfungsi bersama-sama untuk mencapai tujuan tertentu. Suatu sistem mempunyai karakteristik atau sifat tertentu yang mempunyai komponen-komponen *(Components),* batas sistem *(Boundary),* lingkungan luar sistem *(Environments),* penghubung *(Interface),* masukan *(Input),* keluaran *(Output),* pengolah *(Process)* dan sasaran *(Objectives)* atau tujuan *(Goal).* (Asep Muhidin, 2017:150)

II.2.2 Klasifikasi Sistem

Sistem dapat diklasifikasikan dari beberapa sudut pandang, diantaranya adalah sebagai berikut :

1. Sistem diklasifikasikan sebagai sistem abstrak *(Abstract System)* dan sistem fisik *(Phisycal System).* Sistem abstrak adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik. Misalnya sistem teologis, yaitu sistem yang berupa pemikiran-pemikiran hubungan antara manusia dengan Tuhan.
2. Sistem fisik merupakan sistem yang ada secara fisik. Misalnya sistem komputer, sistem akuntansi, sistem produksi dan lain sebagainya.

Sistem diklasifikasikan sebagai sistem alamiah *(Natural System)* dan sistem buatan manusia *(Human Made System)* :

1. Sistem alamiah adalah sistem yang terjadi melalui proses alam, tidak dibuat manusia. Misalnya sistem perputaran bumi.
2. Sistem buatan manusia adalah sistem yang dirancang oleh manusia. Sistem buatan manusia yang melibatkan interaksi antara manusia dengan mesin disebut dengan *Human Machine System* atau ada yang menyebut dengan *Man Machine System*. Sistem informasi merupakan contoh *Man Machine System*, karena menyangkut penggunaan komputer yang berinteraksi dengan manusia.

Sistem diklasifikasikan sebagai sistem tertentu *(Deterministic System)* dan sistem tak tentu *(Probabilistic System)* :

1. Sistem tertentu beroperasi dengan tingkah laku yang sudah dapat diprediksi. Interaksi diantara bagian-bagiannya dapat dideteksi dengan pasti, sehingga

keluaran dari sistem dapat diramalkan. Sistem komputer adalah contoh dari sistem tertentu yang tingkah lakunya dapat dipastikan berdasarkan program-program yang dijalankan.

2. Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas.

Sistem diklasifikasikan sebagai sistem tertutup *(Closed System)* dan sistem terbuka *(Open System)* :

1. Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luarnya. Sistem ini bekerja secara otomatis tanpa adanya turut campur tangan dari pihak diluarnya. Secara teoritis sistem tertutup ini ada, tetapi kenyataannya tidak ada sistem yang benar-benar tertutup, yang ada hanyalah *Relatively Closed* System (secara relatif tertutup, tidak benar-benar tertutup).
2. Sistem terbuka adalah sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. Sistem ini menerima masukan dan menghasilkan keluaran untuk lingkungan luar atau subsistem yang lainnya. Karena sistem sifatnya terbuka dan terpengaruh oleh lingkungan luarnya, maka suatu sistem harus mempunyai suatu sistem pengendalian yang baik. Sistem yang baik harus dirancang sedemikian rupa, sehingga secara relatip tertutup karena sistem tertutup akan bekerja secara otomatis dan terbuka hanya untuk pengaruh yang baik saja (Asep Muhidin, 2017:150).

II.2.3. Sistem Pakar

Sistem Pakar adalah sistem berbasis komputer yang menggunakan pengetahuan, fakta, dan teknik penalaran dalam memecahkan masalah yang biasanya hanya dapat dipecahkan oleh seorang pakar dalam bidang tersebut. Pada dasarnya sistem pakar diterapkan untuk mendukung aktivitas pemecahan masalah, beberapa aktivitas pemecahan yang dimaksud antara lain yaitu pembuatan keputusan, pemaduan pengetahuan, pembuatan desain, perencanaan, prakiraan, pengaturan, pengendalian, diagnosis, perumusan, penjelasan, pemberian nasihat, dan pelatihan (Irpan, Dkk : 2018: 218).

II.2.4. Metode Hybrid Case Based

Metode *hybrid case based* merupakan penggabungan beberapa metode sistem rekomendasi yang bertujuan untuk mengatasi kekurangan dari masing-masing metode. Sistem rekomendasi ini menyaring informasi menggunakan metode *hybrid*. *Rule Based Reasoning* dan *Case Based Reasoning.*

Rule Based Reasoning (*RBR*) merupakan aturan-aturan logis di mana setiap aturannya didapat dari studi literatur dan informasi dari ahli tanpa melihat kasus yang dihadapi. Selain itu ada beberapa cara alternatif untuk memperoleh aturan tersebut menggunakan metode pembelajaran mesin berdasarkan data empiris yang ada. Satu aturan direpresentasikan dengan: IF <kondisi> THEN <kesimpulan>, di mana setiap kondisi-kondisi dari aturan keaturan yang lainnya terhubung satu dengan yang lain melalui penghubung logika seperti penghubung dan, atau, negasi, serta penghubung lainnya membentuk sebuah fungsi logis.

Implementasi *Rule Based Reasoning* akan dilakukan saat pencocokan gejala yang terindikasi gejala kerusakan (Irpan, 2018).

Pada penalaran berbasis kasus *(case based reasoning*), menggunakan pendekatan kecerdasan buatan (*Artificial Intelligent*) yang menitikberatkan pemecahan masalah dengan didasarkan pada *knowledege* dari kasus-kasus sebelumnya. Secara umum, metode ini terdiri dari 4 langkah, yaitu:

1. *Retrieve*

Pada langkah retrieve kasus yang sebenarnya terjadi diambil. Sebuah kasus terdiri dari permasalahan, solusi dan langkah-langkah bagaimana permasalahan dapat dipecahkan. Permasalahan tersebut mempunyai pola dasar untuk memecahkan masalah yang nantinya.

2. *Reuse*

Kasus yang sudah ada digunakan kembali, dengan cara menyesuaikan masalah dengan keadaan yang terjadi saat ini sehingga permasalahan saat ini mendapatkan solusi yang tepat. Setelah solusi diuji dan kurang memuaskan solusi akan direvisi sampai menemukan solusi yang diinginkan pada langkah revise. Pada proses *Reuse* akan menyalin, menyeleksi, dan melengkapi informasi yang akan digunakan.

3. *Revise*

Proses ini informasi tersebut akan dikalkulasi, dievaluasi, dan diperbaiki kembali untuk mengatasi kesalahan-kesalahan yang terjadi pada permasalahan baru.

4. *Retain*

Proses ini dimana kasus akan disimpan bersama dengan solusi dan langkah pengerjaanya. Dengan demikian, jika ada permasalahan yang mirip dengan kasus tersebut, solusinya sudah ditemukan. Implementasi case based reasoning akan dilakukan saat pencocokan gejala mesin yang terindikasi gejala kerusakan. Gejala-gejala tersebut dicocokan dengan kasus-kasus sebelumnya dimana kasus tersebut dikumpulkan sebagai binary point, hasil dari case based reasoning ini akan mengeluarkan output binary yang nantinya akan dilihat kecocokannya antara kerusakan satu dengan kerusakan lainnya.

Untuk mempermudah menganalisis dilakukan perhitungan berdasarkan kasus lama yang merupakan basis pengetahuan yang dimiliki oleh sistem dan ditangani oleh seorang pakar.

Proses *Retrive* merupakan proses pencarian kemiripan kasus baru dengan kasus yang lama. Pencarian kemiripan antara kasus baru dengan kasus lama dilakukan dengan cara mencocokan gejala yang diinputkan oleh pengguna dengan gejala yang ada pada basis pengetahuan. Pada proses *retrieve* ini akan dilakukan pembobotan dengan menggunakan metode *Cosine Similarity*. Metode *Cosine Similarity* merupakan metode yang digunakan untuk menghitung *similarity* (tingkat kesamaan) antar dua buah kasus. Adapun rumus untuk melakukan perhitungan kedekatan antara kasus lama dengan kasus baru sebagai berikut:

Similarity : $(A_i . B_i) = \frac{A_i . B_i}{|A_i| . |B_i|} = \frac{\sum_{i=1}^{n} (A_i . B_i)}{\sqrt{\sum_{i=1}^{n} A_i^n . \sum_{i=1}^{n} = B_i^n}}$

Keterangan :

A = vektor (kasus baru)

B = vektor (kasus lama)

A*i* = *bobot istilah i dalam blok Ai*

Bi = *bobot istilah i dalam blok Bi*

n = jumlah vector.

i = jumlah *istilah* dalam dokumen

Nilai yang dihasilkan oleh Cosine Similarity berkisar antara 0 sampai 1, semakin besar nilainya maka sudut yang dihasilkan kedua vektor semakin kecil, kemakin kecil sudut yang dihasilkan, semakin mirip teks yang dibandingkan.

II.2.5. *Macromedia Dreamweaver 8*

Macromedia Dreamweaver 8 adalah aplikasi desain dan pengembangan web yang menyediakan editor wysiwyg visual (bahasa sehari-hari yang disebut sebagai Design view) dan kode editor dengan fitur standar seperti syntax high lighting, code completion, dan code collapsing serta fitur lebih canggih seperti real-timesyntax checking dan code introspection untuk menghasilkan petunjuk kode untuk membantu pengguna dalam menulis kode. Tata letak tampilan Design memfasilitasi desain cepat dan pembuatan kode seperti memungkinkan pengguna dengan cepat membuat tata letak dan manipulasi elemen HTML.

Dreamweaver memiliki fitur browser yang terintegrasi untuk melihat halaman web yang dikembangkan di jendela pratinjau program sendiri agar konten memungkinkan untuk terbuka di web browser yang telah terinstall. Macromedia Dreamweaver dapat menggunakan ekstensi dari pihak ketiga untuk 15 memperpanjang fungsionalitas inti dari aplikasi, yang setiap pengembang web bisa menulis (sebagian besar dalam HTML dan JavaScript).

Dreamweaver didukung oleh komunitas besar pengembang ekstensi yang membuat ekstensiyang tersedia (baik komersial maupun yang gratis) untuk pengembangan web dari efek rollover sederhana sampai full-featured shopping cart. (Jurnal Web Dhanamas, April 2016:3; Dionne Luhukay, Dr.Yuli Karyanti, Skom, MMSI).

II.2.6. *Database* (Basis Data)

Database atau basis data adalah kumpulan data yang saling berhubungan. Istilah tersebut bisa digunakan pada sistem yang berelasi dan terkomputerisasi. Dalam pengertian umum, database diartikan sebagai gabungan dari elemen-eleman data yang berhubungan dengan teroganisir (Rita dkk, 2018:25-26).

Basis data sendiri dapat didefinisikan dari beberapa sudut pandang yang ada seperti :

1. Himpunan kelompok data arsip yang saling berhubungan dengan terorganisir sedemikian rupa agar kelak dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan/*Redudancy* yang tidak perlu untuk memenuhi berbagai kebutuhan.

3. Kumpulan file/table/arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronik.

Tujuan utama dari pemanfaatan basis data dalam pengolahan data adalah :

1. Kecepatan dan Kemudahan (*Speed*)

   Pemanfaatan basis data memungkinkan kita untuk dapat menyimpan data atau melakukan perubahan atau manipulasi terhadap data atau menampilkan kembali data tersebut dengan lebih cepat dan mudah, daripada jika kita menyimpan data secara manual / non elektronik atau secara elektronik tetapi tidak dalam bentuk penerapan basis data misalnya dalam bentuk *Spread Sheet* atau dokumen teks biasa.

2. Efisiensi Ruang Penyimpanan (*Space*)

   Karena keterkaitan yang erat data dalam sebuah basis data , maka redudansi data pasti terjadi. Banyaknaya redudansi ini tentu akan memperbesar ruang penyimpanan yang disediakan. Dengan basis data, efisiensi atau optimalisasi penggunaan ruang penyimpanan dapat dilakukan, karena kita dapat melakukan penekanan jumlah redudansi data, baik atau dengan menerapkan sejumlah aturan dalam pengkodean atau dengan membuat relasirelasi antar kelompok antar data dan saling berhubungan.

3. Keakuratan (*Accurancy*)

   Pemanfaatan pengkodean atau pembentukan relasi antar data bersama dengan penerapan aturan atau batasan (constraint) tipe data, domain data, keunikan data dan sebagainya sangat berguna untuk menekan ketidakakuratan pemasukan atau penyimpanan data.

4. Ketersediaan (*Availability*)

   Data akan selalu siap dibutuhkan jika sewaktu - waktu akan di butuhkan atau digunakan.

5. Kelengkapan (*Completeness*)

   Kelengkapan data menyimpan struktur yang mendefinisi detail dari tiap objek sehingga kita dapat melakukan perubahan struktur dalam basis data baik dalam penambahan objek baru / table atau dengan penambahan kolom pada suatu table.

6. Keamanan (*Security*)

   Keamanan untuk menentukan user yang boleh menggunakan basis data beserta objek-objek didalamnya dan menentukan jenis-jenis operasi apa saja yang boleh dilakukan user.

7. Kebersamaan Pemakai (*Shareability*)

   Pemakai basis data seringkali tidak terbatas pada satu pemakai saja di satu lokasi atau satu sistem aplikasi saja. Data pegawai dalam basis data kepegawaian, misalkan dapat digunakan oleh banyak sistem (sistem penggajian, sistem akuntansi, sistem inventori, dan lain-lain). Basis data yang dikelola oleh sistem aplikasi yang mendukung lingkungan *Multiuser*, akan dapat memenuhi kebutuhan ini, tapi tetap dengan menjaga atau menghindari munculnya persoalan baru seperti inkonsistensi data karena data yang sama diubah oleh banyak pemakai disaat yang bersamaan atau kondisi *Deadlock* karena ada banyak pemakai yang saling menunggu untuk menggunakan data.

II.2.7. *AppServ*

Appserv salah satu Server Web dalam membangun Website. *Appserv* adalah sebuah aplikasi *Web Server* lokal yang terdiri dari Apache, My SQL, PHP, dan PHP My Admin. *Appserv* merupakan sebuah aplikasi open source yang mendukung sebagai aplikasi untuk dijadikan Web Server. *Appserv* merupakan *Web Server* yang mudah di gunakan yang dapat melayani halaman dinamis. Untuk membangun sebuah *Web Server*, salah satu program yang handal dan gratis yang penulis gunakan dalam membuat tugas akhir ini adalah Appserv-win32-8.5.0. exe. (Rita dkk, 2018:27)

II.2.8. *MySQL*

MySQL adalah sebuah program database server yang mampu menerima dan mengirimkan datanya dengan cepat, multiuser serta menggunakan perintah standar *SQL*. *MySQL* memiliki dua bentuk lisensi, yaitu *Free Software* dan *Shareware*. *MySQL* yang biasa digunakan adalah *MySQL Free Software* yang berada di bawah lisensi *GNU/GPL (General Public License)*. Sebagai *database server* yang *free*, artinya *MySQL* dapat secara bebas digunakan untuk kepentingan pribadi atau usaha. Selain sebagai *server*, *MySQL* dapat juga berperan sebagai *client* sehingga sering disebut *database client/server* (Abdul dkk, 2017:176).

II.2.9. UML *(Unified Modeling Language)*

Unified Modeling Language (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

UML merupakan metodologi dalam mengembangkan system berorientasi Objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem. Dalam membangun perancangan system dengan alat bantu perancangan *UML* ada beberapa tahapan yang akan dilakukan yaitu *Use Case Diagram*, *Class Diagram*, *Activity Diagram* dan *Sequence Diagram* (Gellysa Urva, dkk (2015) :

Dalam skripsi ini, peneliti menggunakan 4 diagram UML sebagai desain sistem yaitu sebagai berikut :

1. *Use Case Diagram*.
2. *Activity Diagram.*
3. *Sequential Diagram.*
4. *Class Diagram.*

1. Use Case Diagram

Use Case Diagram adalah suatu diagram yang membuat bagaimana proses proses dari sistem yang akan diterapkan (dengan cara apa, oleh siapa dan dimana). use case diagram merupakan diagram yang penulis gunakan untuk menjelaskan secara umum proses - proses yang terjadi pada sistem yang dirancang. Simbol-simbol yang digunakan dalam *Use Case Diagram* yaitu:

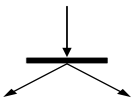
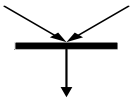
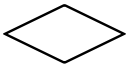
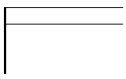
Tabel II.1 Simbol *Use Case Diagram*

Gambar	Keterangan
	*Use Case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktif yang dinyatakan dengan menggunakan kata kerja.
	*Actor* atau Aktor adalah *Abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerjadan tugas-tugas yang berkaitan dengan peran pada kontekstarget sistem. Orang atau sistem bias muncul dalam beberapa peran.
	Asosiasi antara aktor dan *usecase*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang memintainteraksi secara langsung dan bukannya mengindikasikan data.
	Asosiasi antara aktor dan *usecase* yang menggunakan panah terbuka untuk mengindikasikan bila actor berinteraksi secara pasif dengan sistem.
	*Include*, merupakan didalam *use case*lain (*required*) atau pemanggilan *usecase* oleh *usecase* lain,contohnya adalah pemanggilan sebuah fungsi program.
	*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

2. Diagram Aktivitas (*Activity Diagram)*

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis simbol-simbol yang digunakan dalam *activity Diagram* yaitu:

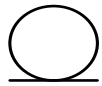
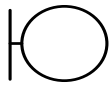
Tabel II.2 Simbol *Activity Diagram*

Gambar	Keterangan
	*Start Point*, diletakkan pada pojok kiri atas dan merupakan awal aktivitas.
	*End Point*, akhir aktivitas.
	*Activities*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork*/percabangan, digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan parallel menjadi satu.
	*Join* (penggabungan) atau *rake*, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan,*true* atau *false.*
	*Swimlane*, pembagian *activity diagram* untuk menunjukkan siapa melakukan apa.

3. Diagram Urutan (*Sequence Diagram)*

Sequence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *Sequence Diagram* yaitu:

Tabel II.3 Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interfaces* atau interaksi Antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan formcetak.
	*Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	Message, symbol mengirim pesan antar *class.*
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true* atau *false.*
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation.*

4. Diagram Kelas (*Class Diagram)*

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem.*Class Diagram* juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint y*ang berhubungan dengan objek yang dikoneksikan. *Class Diagram* secara khas meliputi: Kelas (*Class*), Relasi *Assosiations*, *Generalitation* dan *Aggregation*, attribute (*Attributes*), operasi (*operation/method*) dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau attribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality.*

Tabel II.4 *Multiplicity Class Diagram*

Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimal 4