BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Berdasarkan penelitian yang dilakukan oleh Jayawardanu dan Hansun (2015) mengenai penelitian yang berjudul Rancang Bangun Sistem Pakar Untuk Deteksi Dini Katarak Menggunakan Algoritma C45, Jayawardanu dan Hansun menyimpulkan Sistem pakar untuk mendeteksi adanya penyakit mata katarak secara dini, sudah berhasil dibangun. Sistem ini berbasis *website*, dengan mengimplementasikan algoritma C4.5 dari metode *learning decision tree*.

Berdasarkan penelitian yang dilakukan olehHimawan, dkk (2015) mengenai Diagnosa Tingkat Kesehatan Pasien Menggunakan Metode *DecisionTree* Himawan, dkk menyimpulkan bahwa aplikasi untuk mendiagnosa tingkat kesehatan pasien menghasilkan *information gain atribut* panas sebesar 0,79. Sehingga proses analisa tingkat kesehatan diakibatkan oleh penyakit dahak yang lebih mengarah kepada bronkhitis.

II.2.1. Sistem Pakar

Sistem pakar adalah sistem komputer yang ditujukan untuk meniru semua aspek (*emulates*) kemampuan pengambilan keputusan (*decision making*) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus layaknya seorang pakar untuk memecahkan masalah.Pakar atau ahli (*expert*) didefenisikan sebagai seseorang yang

memiliki pengetahuan atau keahlian khusus yang tidak dimiliki oleh kebanyakan orang. Seorang pakar dapat memecahkan masalah yang tidak mampu dipecahkan kebanyakan orang. Dengan kata lain, dapat memecahkan suatu masalah dengan lebih efisien namun bukan berarti lebih murah. Pengetahuan yang dimuat ke dalam sistem pakar dapat berasal dari seorang pakar atau pun pengetahuan yang berasal dari buku, jurnal, majalah, dan dokumentasi yang dipublikasikan lainnya, serta orang yang memiliki pengetahuan meskipun bukan ahli. Istilah sistem pakar (*expert system*). Sering disinonimkan dengan sistem berbasis pengetahuan (*knowledge-based system*) atau sistem pakar berbasis pengetahuan (*knowledge based expert system*). (Rika Rosnelly, 2012).

Sistem pakar merupakan cabang dari AI yang memiliki *knowledge base.* Sistem pakar dapat bekerja sesuai dengan pengetahuan pakar yang dimasukkan ke dalamnya. Pengetahuan tersebut diproses menggunakan algoritma yang ada untuk menghasilkan *rules. Rules* tersebut dijadikan dasar dalam menentukan hasil akhir dari setiap data yang diuji. (Jayawardanu dan Hansum, 2015).

Sistem pakar adalah salah satu bidang ilmu komputer yang mendayagunakan komputer sehingga dapat berperilaku cerdas seperti manusia. Sistem pakar sebagai sebuah program yang difungsikan untuk menirukan pakar manusia harus bisa melakukan hal-hal yang dapat dikerjakan oleh seorang pakar. (Mariana, dkk , 2012).

II.2.2. Kelebihan Sistem Pakar

Sistem pakar memiliki beberapa fitur menarik yang merupakan kelebihannya, seperti :

1. Meningkatnya ketersediaan (*increased availability*). Kepakaran atau keahlian menjadi tersedia dalam sistem komputer. Dapat dikatakan bahwa sistem pakar merupakan produksi kepakaran secara masal (*massproduction*).
2. Mengurangi biaya (*reduced cost*). Biaya yang diperlukan untuk menyediakan keahlian per satu orang *user* menjadi berkurang.
3. Mengurangi bahaya (*reduced danger*). Sistem pakar dapat digunakan di lingkungan yang mungkin berbahaya bagi manusia.
4. Permanen (*permanence*). Sistem pakar dan pengetahuan yang terdapat di dalamnya bersifat lebih permanen dibandingkan manusia yang dapat merasa lelah, bosan, dan pengetahuannya hilang saat sang pakar meninggal dunia.
5. Keahlian *multiple* (*multiple expertise*). Pengetahuan dari beberapa pakar dapat dimuat ke dalam sistem dan bekerja secara simultan dan kontinyu menyelesaikan suatu masalah setiap saat. Tingkat keahlian atau pengetahuan yang digabungkan dari beberapa pakar dapat melebihi pengetahuan yang digabungkan dari beberapa pakar dapat melebihi pengetahuan satu orang pakar.
6. Meningkatkan kehandalan (*increased reliability*). Sistem pakar meningkatkan kepercayaan dengan memberikan hasil yang benar sebagai

alternatif pendapat dari seorang pakar atau sebagai penengah jika terjadi konflik antara beberapa pakar. Namun hal tersebut tidak berlaku jika sistem dibuat oleh salah seorang pakar, sehingga akan selalu sama dengan pendapat pakar tersebut kecuali jika sang pakar melakukan kesalahan yang mungkin terjadi pada saat tertekan atau stres.

7. Penjelasan (*explanation*). Sistem pakar dapat menjelaskan detail proses penalaran (*reasoning*) yang dilakukan hingga mencapai suatu kesimpulan. Seorang pakar mungkin saja terlalu lelah, tidak bersedia atau tidak mampu melakukannya setiap waktu. Hal ini akan meningkatkan tingkat kepercayaan bahwa kesimpulan yang dihasilkan adalah benar.
8. Respon yang cepat (*fast response*). Respon yang cepat atau *real time* diperlukan pada beberapa aplikasi. Meskipun bergantung pada *hardware* dan *software* yang digunakan, namun sistem pakar relatif memberikan respon yang lebih cepat dibandingkan seorang pakar.
9. Stabil, tidak emosional, dan memberikan respon yang lengkap setiap saat (*steady, unemotional, and complete response at all times*). Karakteristik ini diperlukan pada situasi *real-time* dan keadaan darurat (*emergency*) ketika seorang pakar mungkin tidak berada pada kondisi puncak disebabkan oleh stress atau kelelahan.
10. Pembimbing pintar (*intelligent tutor*). Sistem pakar dapat berperan sebagai *intelligent tutor* dengan memberikan kesempatan pada *user* untuk menjalankan contoh program dan menjelaskan proses *reasoning* yang dilakukan.

Basis data cerdas (*intelligent database*).Sistem pakar dapat digunakan untuk mengakses basis data secara cerdas. (Rika Rosnelly, 2012).

II.2.3. Konsep Umum Sistem Pakar

Pengetahuan yang dimiliki sistem pakar direpresentasikan dalam beberapa cara. Salah satu metode yang paling umum digunakan adalah tipe *rule* menggunakan format *IF THEN*. Banyak sistem pakar yang dibangun dengan mengekspresikan pengetahuan dalam bentuk *rules*. Bahkan, pendekatan berbasis pengetahuan (*knowledgebased approach*) untuk membangun sistem pakar telah mematahkan pendekatan awal yang digunakan pada sekitar tahun 1950-an dan 1960-an yang menggunakan teknik penalaran (*reasoning*) yang tidak mengandalkan pengetahuan. (Rika Rosnelly, 2012).

II.2.4. Elemen Manusia Pada Sistem Pakar

Sistem pakar tidak lepas dari elemen manusia yang terkait di dalamnya. Personil yang terkait dengan sistem pakar ada 4 yaitu :

1. Pakar (*expert*)
2. Pembangun pengetahuan (*knowledge engineer*)
3. Pembangunan sistem (*system engineer*)
4. Pemakai (*user*)

Paling tidak terdapat dua komponen orang atau lebih yang berpartisipasi dalam pembangunan dan penggunaan sistem pakar, yakni sedikitnya seorang pembangun pengetahuan dan seorang pakar. (Rika Rosnelly, 2012).

II.2.4.1. Pakar

Pakar adalah seorang individu yang memiliki pengetahuan khusus, pemahaman, pengalaman, dan metode-metode yang digunakan untuk memecahkan persoalan dalam bidang tertentu. (Rika Rosnelly, 2012)

II.2.4.2. Pembangun/ Pembuat Pengetahuan

Pembuat pengetahuan memiliki tugas utama menerjemahkan dan merepresentasikan pengetahuan yang diperoleh dari pakar, baik berupa pengalaman pakar dalam menyelesaikan masalah maupun sumber terdokumentasi lainnya ke dalam bentuk yang bisa diterima oleh sistem pakar. Dalam hal ini pembangunan pengetahuan (*knowledge engineer*) menginterpretasikan dan merepresentasikan pengetahuan yang diperoleh dalam bentuk jawaban-jawaban atas pertanyaan-pertanyaan yang diajukan pada pakar atau pemahaman, penggambaran analogis, sistematis, konseptual yang diperoleh dari membaca beberapa dokumen cetak seperti *text book*, jurnal, makalah, dan sebagainya. Kurangnya pengalaman *knowledge engineer* merupakan kesulitan utama dalam mengkonstruksi sistem pakar. Untuk mengatasi hal tersebut, perancang sistem pakar menggunakan *tools* komersial. (Seperti pada editor-editor khusus maupun *logic debuggers*) dan usahanya akan dipusatkan pada pembangunan mesin inferensi. (Rika Rosnelly, 2012).

II.2.4.3. Pembangun/ Pembuat Sistem

Pembangunan sistem adalah orang yang bertugas untuk merancang antarmuka pemakai sistem pakar, merancang pengetahuan yang sudah

diterjemahkan oleh pembangun pengetahuan ke dalam bentuk yang sesuai dan dapat diterima oleh sistem pakar dan mengimplementasikannya ke dalam mesin inferensi. Selain hal tersebut, pembangun sistem juga bertanggung jawab apabila sistem pakar akan diintegrasikan dengan sistem komputerisasi lain. Alat pembangun (*tool builder*) dapat dipakai untuk menyajikan atau membangun *tool* yang spesifik. Penjual (*vendor*) dapat memberikan *tool* dan saran, staf pendukung dapat memberikan saran dan bantuan secara teknis dalam proses pembangunan sistem pakar. (Rika Rosnelly, 2012).

II.2.4.4. Konsep Umum Sistem Pakar

Konsep-konsep dasar dari sebuah sistem pakar adalah :

1. Keahlian (*Expertise*)

Keahlian merupakan pengetahuan khusus yang dimiliki oleh seseorang melalui latihan, belajar, serta pengalaman-pengalaman yang dialami pada suatu bidang tertentu dalam jangka waktu yang cukup lama. Dengan pengetahuan tersebut seorang pakar dapat memberikan keputusan yang lebih baik dan cepat dalam menyelesaikan suatu permasalahan yang sulit.

2. Ahli atau pakar (*Expert*)

Seorang pakar harus memiliki kemampuan menyelesaikan permasalahan pada bidang tertentu yang ditanganinya, kemudianmemberikan penjelasan mengenai hasil dan kaitannya dengan permasalahan yang ada. Untuk meniru kepakaran seorang manusia, perlu dibangun sebuah sistem komputer yang menunjukan seluruh karakteristik tersebut. Namun hingga saat ini, pekerjaan

dibidang sistem pakar terfokus pada aktifitas penyelesaian masalah dan memberikan penjelasan mengenai solusinya.

3. Memindahkan Keahlian (*Transfering Expertise*)

Tujuan dari sistem adalah memindahkan keahlian yang dimiliki oleh seorang pakar ke dalam sebuah sistem komputer, kemudian dari sebuah sistem komputer kepada orang lain yang bukan pakar.

4. Kesimpulan (*Inference*)

Keistimewaan dari sistem pakar adalah kemampuannya dalam memberikan saran, yaitu dengan menempatkan keahlian ke dalam basis pengetahuan (*Knowledge Base*) dan membuat program yang mampu mengakses basis pengetahuan sehinggga sistem dapat memberikan kesimpulan. Kesimpulan dibentuk di dalam komponen yang dinamakan mesin pengambil kesimpulan (*Inference Engine*), dimana berisi aturan-aturan untuk menyelesaikan masalah.

5. Aturan (*Rule*)

Umumnya sistem pakar adalah sistem berbasis aturan, yaitu pengetahuan yang terdiri dari aturan-aturan sebagai prosedur penyelesaian masalah. Pengetahuan tersebut digambarkan sebagai suatu urutan seri dari kaidah-kaidah yang sudah dibuat.

6. Kemampuan Penjelasan (*Explanation Capability*)

Keistimewaan lain dari sistem pakar adalah kemampuannya dalam memberikan saran atau rekomendasi serta menjelaskan mengapa tindakan tertentu tidak dianjurkan. Pemberian penerangan dan pendapat ini dilakukan

dalam suatu subsistem yang dinamakan subsistem penjelasan (*explanation subsystem*). (Sinaga dan Sembiring, 2016).

II.2.5. Komponen-Komponen Sistem Pakar

Untuk pembangun sistem, maka komponen-komponen dasar yang minimal harus dimiliki adalah sebagai berikut :

1. Antar muka (*user interface*).
2. Basis pengetahuan (*knowledge base*).
3. Mesin inferensi (*Inference Engine*). (Istiqomah dan Fadlil, 2013).

II.2.6. Karakteristik Sistem Pakar

Sistem pakar umumnya dirancang untuk memenuhi beberapa karakteristik umum berikut ini :

1. Kinerja sangat baik (*high performance*). Sistem harus mampu memberikan respon berupa saran (*advice*) dengan tingkat kualitas yang sama dengan seorang pakar atau melebihinya.
2. Waktu respon yang baik (*adequate respon time*). Sistem juga harus mampu bekerja dalam waktu yang sama baiknya (*reasonable*) atau lebih cepat dibandingkan dengan seorang pakar dalam menghasilkan keputusan. Hal ini sangat penting terutama pada sistem waktu nyata (*real-time*).
3. Dapat diandalkan (*good reliability*). Sistem harus dapat diandalkan dan tidak mudah rusak/ *crash*.
4. Dapat dipahami (*understandable*). Sistem harus mampu menjelaskan langkah-langkah penalaran yang dilakukannya seperti seorang pakar.

5. Fleksibel (*flexibility*). Sistem menyediakan mekanisme untuk menambah, mengubah, dan menghapus pengetahuan. (Rika Rosnelly, 2012 : 21).

II.3. Penyakit Miom Uteri

Mioma uteri adalah tumor jinak otot polos uterus yang terdiri dari sel-sel jaringan otot polos, jaringan pengikat fibroid dan kolagen. Kejadian mioma uteri di Indonesia sebesar 2,39%-11,70% pada semua penderita ginekologi yang dirawat. Faktor-faktor risiko seperti umur, paritas, umur *menarche* dan status haid dapat menyebabkan terjadinya mioma uteri. (Lilyani, dkk, 2012).

Mioma uteri adalah tumor jinak pada uterus. Insidensinya sekitar 20%-30% dari seluruh wanita dan terus mengalami peningkatan. Tumor ginekologi kedua terbanyak di Indonesia. Umumnya ditemukan pada wanita usia reproduksi dan hanya 10% mioma uteri yang masih tumbuh setelah menopause. Kira-kira 60% asimtomatik dan hamper 50% ditemukan secara kebetulan pada pemeriksaan ginekologik. (Sabrianti pasingg, dan Freddy Wagey, 2015).

II.4. Metode C45

Metode C45 merupakan kelompok algoritma*decision tree*. Metode ini mempunyai inputberupa *training samples* dan *samples*. *Training samples* berupa data contoh yang akandigunakan untuk membangun sebuah *tree*yang telah diuji kebenarannya. Sedangkan*samples* merupakan *field-field* data

yangnantinya akan kita gunakan sebagai parameterdalam melakukan klasifikasi data. (Sari dan Sindunata, 2014).

Secara umum algoritma C45 untuk membangunpohon keputusan adalah sebagai berikut :

a. Pilih Atribut sebagai akar.
b. Baut cabang untuk tiap-tiap nilai.
c. Bagi kasus dalam cabang.
d. Ulangi proses untuk setiap cabang sampaisemua kasus pada cabang memiliki kelasyang sama.

Untuk memilih atribut sebagai akar, didasarkanpada nilai *gain* tertinggi dari atribut-atribut yang ada.Untuk menghitung *gain* digunakan ,rumus seperti pada persamaan berikut :

$$Gain\ (S,A) = Entrophy\ (S) \sum_{i=1}^{n} - pi * \log_2 pi \quad \text{.....(1)}$$

Keterangan :

S : Himpunan kasus

A : Atribut

n : Jumlah Partisi Atribut A

|Si| : Jumlah kasus pada partisi ke-i

|S| : Jumlah Kasus dalam S

Sementara itu, penghitungan nilai *entrophy* dapat dilihat pada persamaan berikut :

$$Entrophy\ (S) = \sum_{i=1}^{n} - pi * \log_2 pi \text{.....(2)}$$

Keterangan :

S : Himpunan kasus

A : Fitur

n : Jumlah partisi S

pi : Proporsi dari Si terhadap S.

II.5. Normalisasi

Normalisasi merupakan parameter digunakan untuk menghindari duplikasiterhadap tabel dalam basis data dan jugamerupakan proses mendekomposisikansebuah tabel yang masih memiliki beberapa anomali atau ketidakwajaran sehinggamenghasilkan tabel yang lebih sederhanadan struktur yang bagus, yaitu sebuah tabelyang tidak memiliki *data redundancy* danmemungkinkan *user* untuk melakukan*insert*, *delete*, dan *update* pada baris(*record*) tanpa menyebabkan inkonsistensidata. (Triyono, 2012).

1. *First Normal Form (1 NF)*

   Sudah tidak ada *repeating group* yaitupengulangan yang terjadi pada beberapaatribut atau kolom dalam sebuah tabel,dan juga setiap atribut harus bernilaitunggal. Atribut *multivalued,composite, derive* tidak tunggal. Setiapnilai dari atribut hanya mempunyai nilaitunggal.

2. *Second Normal Form (2 NF)*

   Untuk menjadikan tabel normal tingkatke 2 maka sudah 1NF dan setiap atributyang bukan *primary key* sepenuhnyasecara *fungsional* tergantung padasemua atributpembentuk *primary key*.

3. *Third Normal Form (3 NF)*

Tabel sudah 2NF dan tidak memiliki*transitive dependencies*, *Transitive dependency* adalah ketika ada atributyang secara tidak langsung tergantungpada *primary key* dan atribut tersebutjuga tergantung pada atribut lain yangbukan *primary key*.

4. *Boyce-codd Normal Form (BCNF)*

   Tabel dalam BCNF jika sudah 3NF dansemua *determinants* adalah *candidatekeys*. Perbedaan 3NF dan BCNF adalahuntuk *functional dependency* A B,3NF memperbolehkan ketergantunganada dalam relasi jika B adalah *PrimaryKey* dan A bukan merupakan *candidatekey*. Sedangkan BCNF menuntut untukketergantungan tetap ada dalam relasi,A harus menjadi *candidate key*.

5. *Fourth Normal Form* (4 NF)

   Relasi berada pada bentuk normalkeempat apabila memenuhi syaratBCNF dan tidak mempunyai *multivaluedependency.*

6. *Fifth Normal Form* (5 NF)

   Tabel bentuk normal kelima seringdisebut PJNF (*Projection Join NormalForm*), penyebutan PJNF karena untuksuatu relasi akan berbentuk normalkelima jika tabel tersebut dapat dipecahatau diproyeksikan menjadi beberapatabel dan dari proyeksi-proyeksi itudapat disusun kembali (*join*) menjaditabel yang sama dengan keadaansemula. Jika penyusunan ini tidakmungkin dilakukan dikatakan padarelasi itu terdapat *join dependencies*dan dikatakan bersifat *lossy join*. (Triyono, 2012).

II.6. *Visual Basic* 2010

Visual basic dibuat oleh Microsoft, merupakan salah satu bahasa pemrograman beriorientasi objek yang mudah dipelajari. Selain menawarkan kemudahan, Visual Basic juga cukup andal untuk digunakan dalam pembuatan berbagai aplikasi, terutama aplikasi *database.* Visual basic merupakan bahasa pemrograman *event drive*, di mana program aplikasi yang dapat berupa kejadian atau *event*, misalnya ketika *user* mengklik tombol atau menekan enter. (Prayogi, dkk, 2015).

II.7. Basis Data (*Database*)

Basis data adalah suatu pengorganisasian sekumpulan data yang saling terkait sehingga memudahkan aktivitas untuk memperoleh informasi. Basis data dimaksudkan untuk mengatasi problem pada sistem yang memakai pendekatan berbasis berkas. Untuk mengelola basis data diperlukan perangkat lunak yang di sebut *Database Management System* (DBMS). DBMS adalah perangkat lunak sistem yang memungkinkan para pemakai membuat, memelihara, mengontrol, dan mengakses basis data dengan cara yang praktis dan efisien. DBMS dapat digunakan untuk mengakomodasikan berbagai macam pemakai yang memiliki kebutuhan akses yang berbeda-beda. (Kadir, 2014).

II.8. SQL *Server* 2008

SQL *Server* 2008 adalah sebuahRDBMS (*Relational Database Management System*) yang sangat *powerful* dan telahterbukti kekuatannya

dalam mengolah data.Dalam versi terbarunya ini, SQL *Server* 2008memiliki banyak fitur yang bisa dihandalkanuntuk meningkatkan performa *database*.SQL *Server* 2008 memiliki suatu GUI(*Graphic User Interface*) yang kita gunakanuntuk melakukan aktivitas sehari hariberkaitan dengan *database*,seperti menulisT-SQL, melakukan *backup* dan *restore database*, melakukan *security databas*eterhadap aplikasi, dan sebagainya. Pada GUItersebut kita bisa melakukan *setting*anterhadap SQL *Server* untuk berkerja lebihoptimal.*Setting*an juga bisa dilakukanmenggunakan *script* untuk memudahkan*developer* mengubah *Setting Options* padaSQL *Server* 2008. (Rulsan, 2013).

II.9. *Unified Modeling Language* (*UML*)

Menurut Windu Gata (2013) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (UML). UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

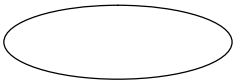
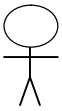
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UMLsaat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem. (Gellysa Urva dan Helmi Fauzi Siregar, 2015, Hal : 93).

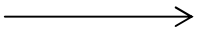
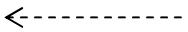
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UMLadalah sebagai berikut:

1. *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram dapat dilihat pada tabel II.1 dibawah ini:

Tabel II.1. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor

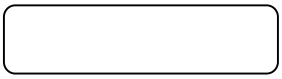
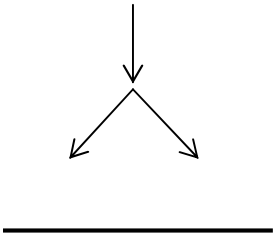
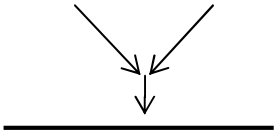
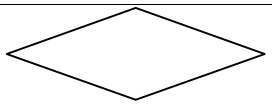
	berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	*Include*, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber:Gellysa Urva dan Helmi Fauzi Siregar; 2015)

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* dapat dilihat pada tabel II.2 dibawah ini:

Tabel II.2. Simbol *Activity Diagram*

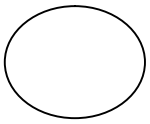
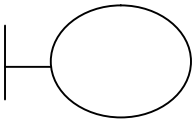
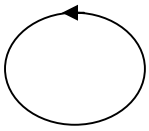
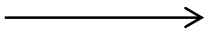
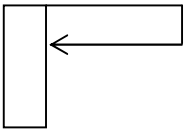
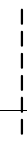
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan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

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015)

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* dapat dilihat pada tabel II.3 dibawah ini :

Tabel II.3. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass*, 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 *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*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*, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*, *activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	*Lifeline*, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.

Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility,* tingkat akses objek eksternal kepada suatu operasi atau atribut.

Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti yang dapat dilihat pada tabel II.4 dibawah ini:

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 maksimum 4

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015)