BAB II

TINJAUAN PUSTAKA

II.1. SistemPakar

Sistem pakar (*Expert System*) mulai dikembangkan pada pertengahan tahun 1960-an oleh *Artificial Intellegence corporation*. Sistem pakar yang muncul pertama kali adalah *General-Purpose Problem Solve (GPS)* yang merupakan sebuah *Predecessor* untuk menyusun langkah-langkah yang dibutuhkan untuk mengubah situasi awal menjadi state tujuan yang telah ditentukan sebelumnya dengan menggunakan domain masalah yang kompleks. (Jusniwati, 2013 : 67).

Sistem pakar adalah computer cerdas yang menggunakan prosedur pengetahuan dan kesimpulan untuk menyelesaikan masalah yang cukup sulit yang memerlukan keahlian manusia dalam permasalahan kehidupan yang nyata. (SulaksonodanDarsono, 2015 : 3).

Secara umum, system pakar adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer yang dirancang untuk memodelkan kemampuan menyelesaikan masalah seperti yang layaknya seorang pakar. Selanjutnya, menurut Martin dan Oxman, system pakar adalah system berbasis komputer yang menggunakan pengetahuan, fakta dan teknik penalaran dalam memecahkan masalah yang biasanya hanya dapat dipecahkan oleh seorang pakar dalam bidang tersebut. Penyusunannya system pakar harus malakukan kombinasi terhadap kaidah-kaidah untuk menarik kesimpulan atau*i nference rules* dengan basis pengetahuan tertentu yang diberikan oleh satu atau lebih pakar dalam bidang

tertentu. Kombinasi dari kedua hal tersebut disimpan dalam komputer, yang selanjutnya digunakan dalam proses pengambilan keputusan untuk penyelesaian masalah tertentu. (Belutowe, 2015 : 52).

II.1.1. KonsepUmumSistemPakar

Konsep-konsep dasar dari sebuah system 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. Ahliataupakar (*Expert*)

Seorang pakar harus memiliki kemampuan menyelesaikan permasalahan pada bidang tertentu yang ditanganinya, kemudian memberikan penjelasan mengenai hasil dan kaitannya dengan permasalahan yang ada. Untuk meniru kepakaran seorang manusia, perlu dibangun sebuah system komputer yang menunjukan seluruh karakteristik tersebut. Namun hingga saat ini, pekerjaan dibidang system pakar terfokus pada aktifitas penyelesaian masalah dan memberikan penjelasan mengenaisolusinya.

3. Memindahkan Keahlian (*Transfering Expertise*)

Tujuan dari system adalah memindahkan keahlian yang dimiliki oleh seorang pakar ke dalam sebuah system komputer.

4. Kesimpulan (*Inference*):

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

5. Aturan (*Rule*)

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

6. KemampuanPenjelasan (*Explanation Capability*)

Keistimewaan lain dari system 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 sub system penjelasan (*explanation subsystem*). (SinagadanSembiring, 2016 : 96).

II.2. *Teorema Bayes*

Teorema bayes merupakan satumetode yang digunakan untuk menghitung ketidakpastian data menjadi data yang pasti dengan membandingkan antara data ya dan tidak. Probabilitas bayes merupakan salah satu cara untuk mengatasi ketidakpastian data. (HarijantodanLatif, 2016 : 177).

Rumus *Teorema Bayes* dapat dilihat sebagai berikut :

$$P(H|E) = \frac{P(E|H) * P(H)}{P(E)}$$

Dimana :

P(H | E) = probabilitashipotesis H jikadiberikan*evidence* E.

P(E | H) = probailitasmunculnya*evidence* E jikadiketahuihipotesis H.

P(H) = probabilitas H tanpamengandung*evidence* apapun.

P(E) = probabilitas*evidence* E.

II.3. *Naïve Bayes*

Naïve Bayes merupakan pengklasifikasian dengan metode probabilitas dans tatistik yang dikemukakan oleh ilmu waninggris Thomas Bayes, yaitu memprediksi peluang di masa depan berdasarkan pengalaman di masa sebelumnya. Metode *Naïve Bayes* juga dinilai berpotensi baik dalam mengklasifikasikan dokumen dibandingkan dengan metode pengklasifikasian lain dalam hal akurasi danefisiensi komputasi. (NurlelahdanWajhillah, 2016 : 53).

$$P(X|Y) = \frac{P(X \cap Y)}{P(Y)}$$

Dimana :

P(X|Y) = probabilitashipotesisXjikadiberikan*evidence* Y.

P(X ∩ Y) = probailitashipotesisXatauhipotesisY.

P(Y) = probabilitasYtanpamengandung*evidence* apapun.

II.4. Penyakit*Crohn*

Penyakit *Crohn* merupakan kondisi peradang ansaluran cernakronik dan idiopatik. Tapi kondisi ini lebih sering terjadi pada bagian akhir usus kecil (ileum) atau usus besar. Penyebab pasti penyakit *Crohn* tidak diketahui. Ini adalah suatukondisi yang terjadi ketika system kekebalan tubuh anda keliru menyerang dan menghancurkan jaringan tubuh yang sehat (gangguanautoimun). Namun penyakit ini dapat diketahui daribeberapa gejala seperti tubuh merasa lemas, denyut nadi lemah atau pun terlalu cepat, mengalami sakit perut yang parah dan lain sebagainya. (Firmansyah, 2013 : 247).

II.5. *Database*

Database adalah sekumpulan tabel-tabel yang saling berelasi, relasi tersebut bias ditunjukkan dengan kunci dari tiap tabel yang ada. Satu *database* menunjukkan satu kumpulan data yang dipakai dalam satu lingkup perusahaan atau instansi. (UrvadanSiregar, 2015, Hal : 93).

II.6. Normalisasi

Normalisasi merupakan para meter digunakan untuk menghindarari duplikasi terhadap table dalam basi data dan juga merupakan proses mendekomposisikan sebuahtabel yang masih memiliki beberapa anomaly sehingga menghasilkant abel yang lebih sederhana dan struktur yang bagus.

1. *First Normal Form (1 NF)*

Sudah tidak ada *repeating group* yaitu pengulangan yang terjadi pada beberapa atribut atau kolom dalam sebuah tabel, dan juga setiap atribut harus bernilai tunggal.

2. *Second Normal Form (2 NF)*

Untuk menjadikan tabel normal tingkatke 2 maka sudah 1NF dan setiap atribut yang bukan *primary key* sepenuhnya secara *fungsional* tergantung pada semua atribut pembentuk *primary key*.

3. *Third Normal Form (3 NF)*

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

4. *Fourth Normal Form* (4 NF)

Relasi berada pada bentuk normal keempat apabila memenuhi syarat BCNF dan tidak mempunyai *multivalue dependency.*

5. *Fifth Normal Form* (5 NF)

Tabel bentuk normal kelima sering disebut PJNF (*Projection Join Normal Form*), penyebutan PJNF karena untuk suatu relasiakan berbentuk normal kelimaji kata belter sebut dapat dipecah atau diproyeksikan menjadi beberapa table dan dari proyeksi-proyek siitu dapat disusun kembali (*join*) menjadi tabel yang sama dengan keadaan semula. (Triyono, 2012 : 20).

II.7. *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 : 3).

II.8. SQL *Server* 2008

SQL *Server* 2008 adalah sebuah RDBMS (*Relational Database Management System*) yang sangat *powerful* dan telah terbukti kekuatannya dalam mengolah data. Dalam versi terbarunya ini, SQL *Server* 2008memiliki banyak fitur yang bias dihandalkan untuk meningkatkan performa *database*. SQL *Server* 2008 memiliki suatu GUI(*Graphic User Interface*) yang kita gunakan untuk melakukan aktivitas sehari hari berkaitan dengan *database*, seperti menulis T-SQL, melakukan *backup* dan *restore database*, melakukan *security database* terhadap aplikasi, dan sebagainya. Pada GUI tersebut kita bias melakukan *setting*an terhadap SQL *Server* untuk berkerja lebih optimal. *Setting*an juga bias dilakukan menggunakan *script* untuk memudahkan *developer* mengubah *Setting Options* pada SQL *Server* 2008. (Rulsan, 2013 : 39).

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.

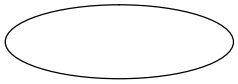
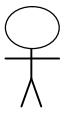
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem. (UrvadanSiregar,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

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah 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*

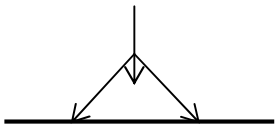
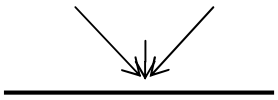
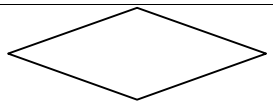
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi 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:UrvadanSiregar; 2015, Hal : 94)

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*

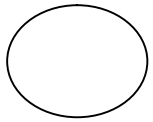
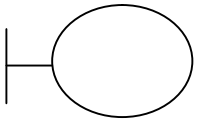
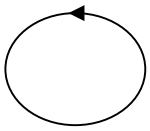
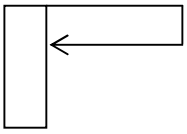
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan 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 : UrvadanSiregar; 2015, Hal : 94)

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 : UrvadanSiregar; 2015, Hal : 95)

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 : UrvadanSiregar; 2015, Hal : 95)