BAB II
TEORI PENUNJANG

2.1 Sistem Pakar

Sistem pakar adalah sistem berbasis komputer yang menggunakan pengetahuan, fakta dan teknik aktivitas cerdas untuk mengatasi masalah yang biasanya dapat diatasi hanya oleh seorang pakar pada bidang tertentu (Martin dan Oxman, 1988).

Secara umum, sistem pakar *(expert system)* adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli. Menurut Durkin, sistem pakar adalah suatu program komputer yang dirancang untuk memodelkan kemampuan penyelesaian masalah yang dilakukan oleh seorang pakar. Sistem pakar yang baik dirancang agar dapat menyelesaikan suatu permasalahan tertentu dengan meniru kerja dari para ahli. Dengan sistem pakar ini, orang awampun dapat menyelesaikan masalah yang cukup rumit yang sebenarnya hanya dapat diselesaikan dengan bantuan para ahli. Bagi para ahli, sistem pakar ini juga akan membantu aktivitasnya sebagai asisten yang sangat berpengalaman.

2.1.1 Konsep Dasar Sistem Pakar

Menurut Efraim Turban, konsep dasar sistem pakar mengandung: keahlian, ahli, pengalihan keahlian, inferensi, aturan dan kemampuan menjelaskan.

Salah satu fitur yang harus dimiliki oleh sistem pakar adalah kemampuan untuk menalar. Jika keahlian-keahlian sudah tersimpan sebagai basis pengetahuan dan sudah tersedia program yang mampu mengakses basisdata, maka komputer harus dapat di program untuk membut inferensi. Proses inferensi ini dikemas dalam bentuk motor interferensi *(inference engine).* Sebagian besar sistem pakar komersial dibuat dalam bentuk *rule-based systems,* yang mana pengetahuan

disimpan dalam bentuk aturan-aturan. Aturan tersebut biasanya berbentuk IF-THEN. Fitur lainnya dari sistem pakar adalah kemampuan untuk merekomendasi. Kemampuan inilah yang membedakan sistem pakar dengan sistem konvensional.

2.1.2 Komponen sistem pakar

Sebuah sistem pakar terdiri atas komponen-komponen sebagai berikut :

1. Basis Pengetahuan

Basis pengetahuan adalah inti program sistem pakar. Basis pengetahuan merupakan tempat penyimpanan pengetahuan dalam memori komputer, dimana pengetahuan ini diambil dari pengetahuan pakar . Pengetahuan ini dapat berasal dari pakar, jurnal, makalah, dan sumber pengetahuan lain.

2. Mesin Inferensi *(Inference Engine)*

Mesin inferensi merupakan otak dari aplikasi sistem pakar. Bagian inilah yang menuntun user untuk memasukkan fakta sehingga diperoleh suatu kesimpulan. Apa yang dilakukan oleh mesin inferensi ini didasarkan pada pengetahuan yang ada dalam basis pengetahuan. Ada dua teknik penalaran *(inferensi)* yaitu penalaran maju *(forward chaining)* dan penalaran mundur *(backward chaining)*.

3. Basis Data (*Database*)

Basis data adalah bagian yang mengandung semua fakta-fakta, baik fakta awal pada saat sistem mulai beroperasi maupun fakta-fakta yang didapatkan pada saat pengambilan kesimpulan sedang dilaksanakan dalam prakteknya

Berikut adalah dua elemen dasar basis data:

1.Fakta, situasi masalah dan teori yang terkait,

2.Heuristic khusus atau *rules*, yang langsung menggunakan pengetahuan untuk menyelesaikan masalah khusus.

4. Antarmuka (*User Interface)*

User interface digunakan untuk media komunikasi antara *user* dengan sistem yang dirancang. Sistem yang mudah dimengerti oleh *user* dapat memudahkan user dalam berkomunikasi dengan sistem.

2.1.3 Ciri-ciri sitem pakar

Sistem pakar yang baik harus memenuhi ciri-ciri sebagai berikut :

1. Memiliki fasilitas informasi yang handal.

2. Mudah dimodifikasi.

3. Dapat digunakan dalam berbagai jenis komputer.

4. Memiliki kemampuan untuk belajar beradaptasi.

2.1.4 Motor Inferensi (*Inference*)

Dalam melakukan inferensi diperlukan adanya proses pengujian kaidah-kaidah dalam urutan tertentu untuk mencari yang sesuai dengan kondisi awal atau kondisi yang berjalan yang sudah dimasukkan pada basis data. Ada 2 cara yang dapat dikerjakan dalam melakukan inferensi, yaitu :

1. Runut Maju *(Forward Chaining)*

Runut maju merupakan proses perunutan yang dimulai dengan menampilkan kumpulan data atau fakta yang meyakinkan menuju konklusi akhir. Runut maju bisa juga disebut sebagai penalaran *forward (forward reasoning)* atau pencarian yang dimotori data *(data driven search)*. Jadi dimulai dari premis-premis atau informasi masukan *(if)* dahulu kemudian menuju konklusi *(then)* atau dapat dimodelkan sebagai berikut:

IF (informasi masukan)

THEN (konklusi)

Informasi masukan dapat berupa data, bukti, temuan, atau pengamatan. Sedangkan konklusi dapat berupa tujuan, hipotesa, penjelasan, atau

diagnosis. Sehingga jalannya penalaran runut maju dapat dimulai dari data menuju tujuan, dari bukti menuju hipotesa, dari temuan menuju penjelasan, atau dari pengamatan menuju diagnosa.

2. Runut Balik *(Backward Chaining)*

Runut balik merupakan proses perunutan yang arahnya kebalikan dari runut maju. Proses penalaran runut balik dengan tujuan/*goal* kemudian merunut balik ke jalur yang akan mengarahkan ke *goal* tersebut, mencari bukti-bukti bahwa bagian kondisi terpenuhi. Jadi secara umum runut balik itu diaplikasikan ketika tujuan atau hipotesis yang dipilih itu sebagai titik awal penyelesaian masalah. Runut balik dimodelkan sebagai berikut:

Tujuan,

IF (kondisi) .

Pada metode runut balik, proses internal selalu mengecek konklusi terlebih dahulu, sebagai praduga awal, baru kemudian mengecek gejala-gejala dipenuhi user atau tidak, bila keseluruhan gejala terpenuhi, maka praduga sistem benar dan dikeluarkan sebagai *output*, bila ada gejala yang tidak terpenuhi berarti praduga sistem salah, selanjutnya sistem akan mengecek konklusi berikutnya.

2.2 Android

Android adalah sistem operasi *Mobile Phone* berbasiskan linux. Android adalah salah satu produk besutan dari android inc. yang telah diakuisisi oleh google.inc yang kemudian dikembangkan menjadi *open source*. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam piranti bergerak. Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Services* (GMS) dan kedua adalah yang benar–benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD). Perkembangan sistem operasi android :

1. Android versi 1.1
2. Android versi 1.5 *(Cupcake)*
3. Android versi 1.6 *(Donut)*
4. Android versi 2.0/2.1 *(Eclair)*
5. Android versi 2.2 *(Frozen Yoghurt)*
6. Android versi 2.3 *(Gingerbread)*
7. Android versi 3.0/3.1 *(Honeycomb)*
8. Android versi 4.0 *(Ice Cream Sandwich)*
9. Android versi 4.1 *(Jelly Bean)*
10. Android versi 4.4 *(KitKat)*

2.3 Penyakit Bayi

Dibandingkan dengan orang dewasa, bayi lebih rentan terhadap panyakit. Kondisi geografis di suatu wilayah menjadikan variasi mikrooranisme penyebab penyakit lebih beragam. Beberapa penyakit mungkin dianggap ringan dan biasa, namun orang tua patut waspada karena penyakit yang dianggap ringan pada bayi dapat menyebabkan dampak yang berbahaya pada bayi tersebut. Diperlukan pengetahuan terhadap penyakit-penyakit yang biasa terjadi pada bayi, agar orang tua dapat bertindak cepat dan tepat dalam mencegah dan menaggulangi kondisi tersebut.

2.4 OOP (*Object Oriented Programming*)

OOP atau *Object Oriented Programming* adalah teknik pemrograman berbasis *object*. Pemrograman dibagi dalam *object-object* kemudian di dalam *object-object* ini ada *property* dan *method*. *Property* ini lebih dikenal dengan *variable* yang bersifat *public* dan *method* ini merupakan *function*. Dengan pengelompokkan-pengelompokkan ini maka *programming* akan terstruktur dengan baik. Bahasa *programming* yang sudah mengimplementasikan OOP adalah C++ dan java. Pemrograman PHP akhir-akhir ini juga bisa dikembangkan menjadi bahasa pemrograman berbasis OOP (*Object Oriented Programming)* dengan lahirnya *framework-framework* berbasis PHP.

2.5 UML *(Unified Modeling Languange)*

UML merupakan kepanjangan dari *Unified Modeling Language* merupakan sebuah bahasa yang berdasarkan grafik / gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan software berbasis orientasi objek *(object-oriented).*

Jenis-jenis diagram UML:

1. Diagram Struktur

Berikut adalah diagram yang termasuk kedalam diagram struktur:

1. Diagram Kelas
2. Diagram Obyek
3. Diagram Komponen
4. Diagram Deployment

2. Diagram Perilaku

Berikut adalah diagram yang termasuk kedalam diagram perilaku:

1. Diagram Usecase
2. Diagram Urutan/sequence

3. Diagram Kolaborasi

4. Diagram Statechart

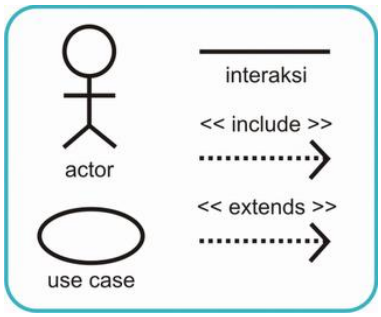
5. Diagram Aktivitas

Dalam pembuatan sistem pakar ini dirancang menggunakan diagram *Use Case, Class, Sequence, Collaboration.*

2.5.1 Diagram *Use Case*

Diagram *use case* digunakan untuk mengambarkan interaksi antara pengguna sistem *(actor)* dengan kasus *(use case)* yang disesuaikan dengan langkah-langkah *(scenario)* yang telah ditentukan. *Use case* diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem, yang ditekankan adalah apa yang diperbuat sistem, dan bukan bagaimana.

Komponen pembentuk *Use Case* diagram:

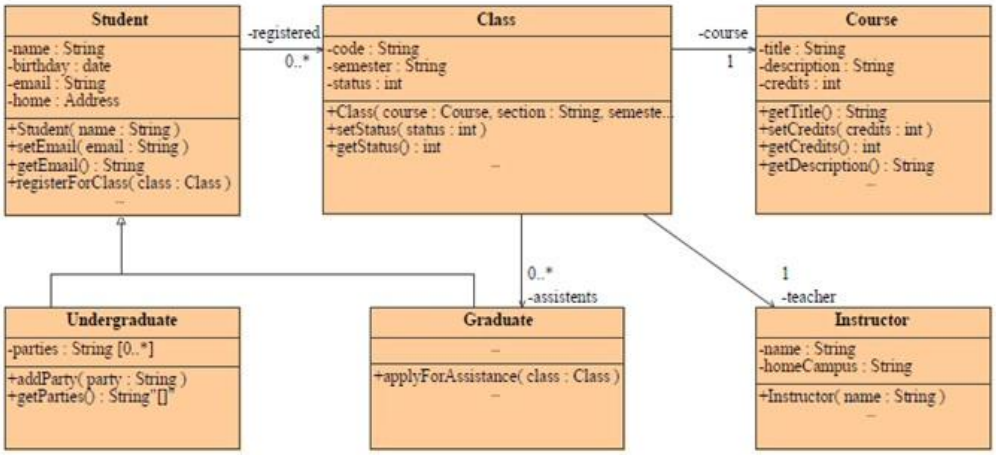


Gambar 2.1 Komponen Pembentuk *Use Case* Diagram

2.5.2 Diagram *Class*

Diagram *class* sangat membantu dalam visualisasi struktur *class* dari suatu sistem. Hal ini disebabkan karena *class* adalah deskripsi kelompok objek-objek dengan properti, perilaku (operasi) dan relasi yang sama. Disamping itu diagram *class* bisa memberikan pandangan global atas sebuah sistem. Hal tersebut tercermin dari *class-class* yang ada dan relasinya satu sama lainnya. Itulah sebabnya diagram *class* menjadi paling popular di UML.

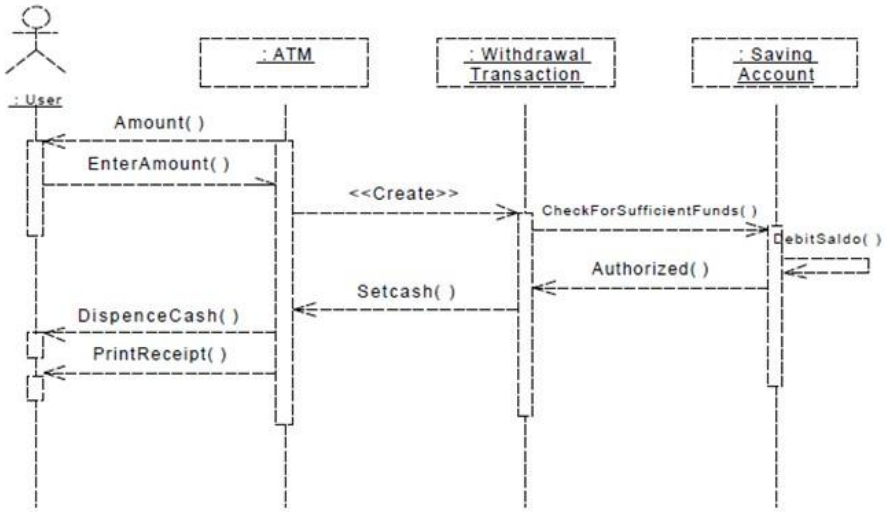
Class dalam notasi UML digambarkan dengan kotak. Nama *class* menggunakan huruf besar di awal kalimatnya dan diletakkan di atas kotak. Bila *class* mempunyai nama yang terdiri dari dua suku kata atau lebih, maka semua suku kata digabungkan tanpa spasi dengan huruf awal tiap suku kata menggunakan huruf besar. Terdapat tiga bagian dalam suatu *class* yaitu nama *class,* atribut dan *operation.* Berikut ini contoh gambar dari diagram *class*:



Gambar 2.2 Contoh Diagram *Class*

2.5.3 Diagram *Sequence*

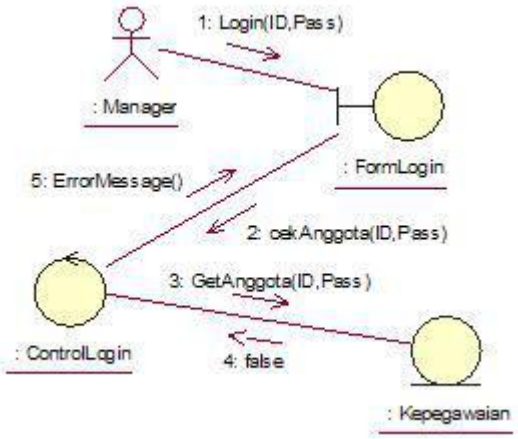
Diagram *sequence* menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. Diagram *sequence* terdiri atar dimensi *vertikal* (waktu) dan dimensi *horizontal* (objek-objek yang terkait). Diagram *sequence* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah event untuk menghasilkan output tertentu. Diawali dari apa yang menggerakkan aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan. Berikut ini adalah contoh gambar diagram *sequence:*



Gambar 2.3 Contoh Diagram *Sequence*

2.5.4 Diagram *Collaboration*

Diagram *collaboration* adalah perluasan dari objek diagram. Objek diagram menunjukkan objek-objek dan hubungannya satu dengan yang lain. Diagram c*ollaboration* menunjukkan *message-message* objek yang dikirimkan satu sama lain. Berikut ini adalah contoh gambar diagram *collaboration:*



Gambar 2.4 Contoh Diagram *Collaboration*

2.6 PHP

PHP adalah kependekan dari *PHP Hypertext Preprocessor,* bahasa interpreter yang mempunyai kemiripan dengan bahasa C dan perl yang mempunyai kesederhaan dalam perintah. PHP merupakan salah satu bahasa

(sering disebut script) pemrograman yang sering digunakan pada sisi *server*. Dengan PHP anda dapat menggunakan *database* MySQL untuk menyimpan data dan mengambilnya kembali pada saat diperlukan.

PHP baik digunakan sebagai pengantar ke bahasa pemrograman *server* side. Pada awalnya PHP adalah singkatan dari *personal Homepage* yang kemudian menjadi PHP *Hypertext Preprocessor,* yang dikembangkan pada tahun 1994 oleh Rasmus Lerdorf sebagai suatu bahasa *server* untuk *personal homepage* yang dimilikinya.

PHP adalah bahasa pemrograman yang dirancang untuk mudah diletakkan di dalam kode HTML. Banyak dijumpai kode PHP yang menyatu dengan kode HTML. Kode PHP diawali dengan tag **<?php** dan diakhiri dengan tag **?>**. Berikut ini contoh kode PHP yang sangat sederhana.

```
<?php

    Echo “Hello World” ;

?>
```

Perintah echo di dalam PHP berguna untuk mencetak nilai, baik teks maupun numerik. Setiap perintah atau statement di dalam kode PHP harus diakhiri dengan tanda titik koma atau *semicolon* (;).

2.6.1 Tipe Data

PHP mendukung delapan tipe nilai atau tipe data. Empat nilai scalar (*single value)* yaitu *integer, floating point, string, dan boolean*, dua nilai koleksi yaitu *array* dan objek serta tipe khusus yaitu *resource* dan *null.*

2.6.2 Variabel

Variabel adalah suatu pengenal di dalam program yang merepresentasikan nilai atau data tertentu. Dengan memasukkan nilai ke dalam variable, maka dapat mengacu variabel tersebut dari tempat lain dalam kode program yang di buat. Dalam PHP, variabel didefinisikan dengan cara membubuhkan tanda $ (*dollar)* di

depannya. Berikut ini langkah-langkah yang diperlukan untuk membuat variabel di dalam PHP:

1. Tulis nama variabel yang di butuhkan:

username

usia

2. Tempatkan tanda $ di depan nama variabel tersebut:

$username

$usia

3. Gunakan tanda sama dengan (=) setelah nama untuk memasukkan nilai literal ke dalam variabel tersebut:

$username = “pras” ;

$usia = 27 ;

Berbeda dengan bahasa pemrograman lain pada umumnya, dalam PHP variabel tidak perlu dideklarasikan terlebih dahulu.

2.7 MySQL

MySQL termasuk dalam kategori *database management system*, yaitu suatu *database* yang terstruktur dalam pengolahan dan penampilan datanya. My SQL merupakan *database* yang bersifat *client server*, di mana data diletakkan di *server* yang bisa diakses melalui komputer *client*. Pengaksesan dapat dilakukan apabila komputer terhubung dengan server.

MySQL dapat juga dikatakan sebagai *Relational Database Management Sistem* (RDBMS), yaitu hubungan antar tabel yang berisi data-data pada suatu *database*. Dengan demikian dapat mempercepat pencarian suatu kata. Tabel-tabel tersebut di-link oleh suatu relasi yang memungkinkan kombinasi data dari beberapa table ketika *user* menginginkan tampilnya informasi dari suatu *database*. Setiap RDBMS pasti memiliki *tool* yang dapat digunakan untuk mempermudah

pengoperasian *database.* MySQL memiliki *tool* atau aplikasi yang disebut PhpMyAdmin.

2.7.1 PhpMyAdmin

Setiap RDMS (*Relational Database Management System)* seperti Oracle, SQL server, MySQL dan lain-lain, pasti memiliki *tool* yang dapat digunakan untuk mempermudah pengoperasian *database*. Oracle memiliki TOAD. SQL *server* memiliki Enterprise Manager dan SQL Query Analyzer. Sedangkan MySQL memiliki *tool* atau aplikasi yang disebut PhpMyAdmin.

PhpMyAdmin merupakan aplikasi berbasiskan *web* yang dikembangkan menggunakan bahasa pemrograman PHP. Melalui PhpMyAdmin, *user* dapat melakukan perintah query tanpa harus mengetikan seperti pada MS DOS. Perintah tersebut misalnya administrasi *user* dan privilages, export dan import *database*, manajemen *database*, manajemen table dan struktur table, dan sebagainya. PhpMyAdmin sangat *user* friendly, sehingga mudah untuk digunakan walaupun pengguna baru.

2.8 ERD *(Entity Relationship Diagram)*

Model *Entity Relationship* diperkenalkan pertama kali oleh P.P. Chen pada tahun 1976. Model ini dirancang untuk menggambarkan persepsi dari pemakai dan berisi objek-objek dasar yang disebut *entity* dan hubungan antar *entity-entity* tersebut yang disebut *relationship*. Pada model ER ini semesta data yang ada dalam dunia nyata ditransformasikan dengan memanfaatkan perangkat konseptual menjadik sebuah diagram, yaitu diagram ER *(Entity Relationship).*

Diagram *Entity-Relationship* melengkapi penggambaran grafik dari struktur logika . Dengan kata lain Diagram E-R menggambarkan arti dari aspek data seperti bagaimana *entity-entity*, atribut-atribut dan *relationship-relationship* disajikan. Sebelum membuat Diagram E-R , tentunya kita harus memahami betul data yang diperlukan dan ruang lingkupnya. Di dalam pembuatan diagram E-R perlu diperhatikan penentuan sesuatu konsep apakah merupakan suatu *entity*, atribut atau *relationship*. Notasi-notasi simbolik di dalam ERD yang dapat kita gunakan adalah:

- Persegi panjang, menyatakan himpunan entitas.
- Lingkaran/elip, menyatakan atribut (atribut yang berfungsi sebagai *key* digarisbawahi).
- Belah ketupat, menyatakan himpunan relasi.
- Garis, sebagai penghubung antara himpunan relasi dengan himpunan entitas dan himpunan entitas dengan atributnya.
- Kardinalitas relasi dapat dinyatakan dengan banyaknya garis cabang atau dengan pemakaian angka (1 dan 1 untuk relasi satu ke satu, 1 dan N untuk relasi satu-ke-banyak atau N dan N untuk relasi banyak-ke-banyak).

2.9 AppsGeyser

AppsGeyser adalah *platform* web gratis yang memungkinkan mengubah setiap konten web menjadi aplikasi android. Selain membuat aplikasi, Appsgeyser juga bisa digunakan untuk mendistibusikan aplikasi yang di buat di google play serta mengukur nilai jual aplikasi yang di buat. Dibangun untuk membantu orang-orang menuangkan ide-ide mereka ke dalam aplikasi android. Pengguna android lebih suka menggunakan aplikasi bukan browser untuk mendapatkan informasi atau menggunakan layanan, karena lebih mudah dan lebih berguna. Dengan AppsGeyser kita dapat dengan cepat membuat sendiri aplikasi android dari konten web yang ada.