

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

2.1.1 Teori Pengembangan Pakar

a. Sistem Pakar

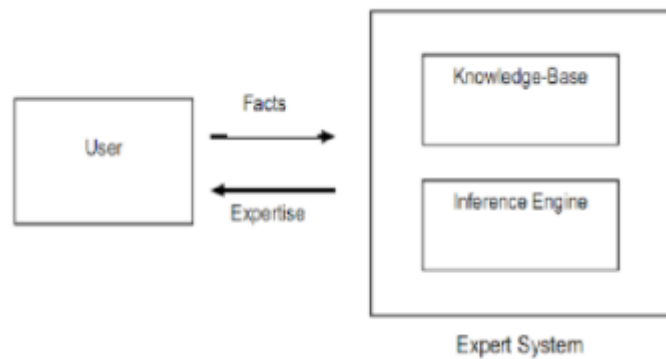
Sistem pakar merupakan salah satu dari beberapa domain masalah atau area dari *Artificial Intelligence* (AI). Menurut Profesor Edward Feigenbaum dalam Rosnelly (2012:2) mendefinisikan sistem pakar, “Sebuah program computer pintar (*intelligent computer program*) yang memanfaatkan pengetahuan (*knowledge*) dan prosedur inferensi (*inference procedure*) untuk memecahkan masalah yang cukup sulit hingga membutuhkan keahlian khusus dari manusia.”

Dengan kata lain, Rosnelly (2012:2) menyimpulkan bahwa, “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 selayaknya seorang pakar memecahkan masalah.”

Menurut Rosnelly (2012:3) menjelaskan pengertian pakar bahwa:

Pakar adalah seseorang dengan pengetahuan dan keahlian khusus dan tidak dimiliki oleh kebanyakan orang. Pengetahuan yang dapat dimuat kedalam sistem pakar dapat berasal dari seorang pakar, atau pengetahuan yang berasal dari buku, jurnal, majalan, dan dokumentasi yang dipublikasikan lainnya, serta seorang yang memiliki pengetahuan meskipun bukan ahli.

Konsep dasar fungsi sistem pakar berbasis pengetahuan, bisa dilihat pada gambar II.1 berikut ini:



Sumber: Rosnelly (2012:4)

Gambar II.1
Konsep Dasar Fungsi Sistem Pakar Berbasis Pengetahuan

b. Struktur Dasar Sistem Pakar

Ada beberapa komponen dalam struktur dasar sistem pakar menurut

Rosnelly (2012:14) yaitu sebagai berikut:

1. Basis Pengetahuan (*knowledge based*)

Berisi pengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah berdasarkan fakta dan aturan yang ada. Pada struktur ini, sistem menyimpan pengetahuan dari pakar berupa rule atau aturan (*if <kondisi> then <aksi>* atau dapat juga disebut *condition-action rules*)

2. Mesin inferensi (*inference engine*)

Merupakan otak dari sistem pakar atau disebut juga dengan *control structure* (struktur kontrol) atau *rule interpreter* yang mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah dengan mencocokkan bagian kondisi dari *rule* yang tersimpan di dalam *knowledge base* dengan data yang tersimpan di *working memory*.

3. *Working Memory*

Menyimpan fakta yang dihasilkan oleh *inference engine* dengan penambahan parameter berupa derajat kepercayaan atau dapat juga dikatakan sebagai *global database* dari fakta yang digunakan oleh *rule-rule* yang ada.

4. *Explanation Facility*

Menyediakan kebenaran dari solusi yang dihasilkan kepada *user (reasoning chain)*

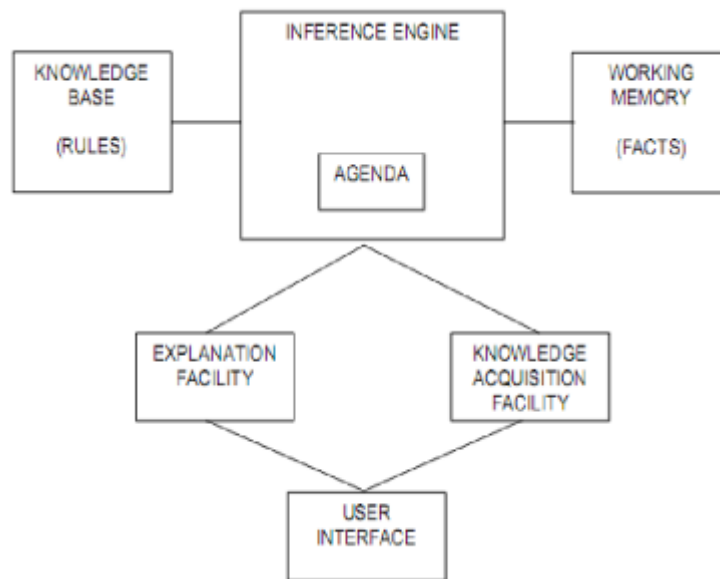
5. *Knowledge Acquisition Facility*

Meliputi proses pengumpulan, pemindahan dan perubahan dari kemampuan pemecahan masalah seorang pakar atau sumber pengetahuan terdokumentasi ke program komputer, yang bertujuan untuk memperbaiki atau mengembangkan basis pengetahuan.

6. *User Interface*

Mekanisme untuk memberi kesempatan kepada user dan siste pakar untuk berkomunikasi, dimana antar muka menerima informasi dari pemakai dan mengubahnya ke dalam bentuk yang dapat diterima oleh sistem.

Struktur dasar sistem pakar juga dapat dilihat pada gambar II.2:



Sumber: Rosnelly (2012:13)

Gambar II.2
Struktur Sistem Pakar

c. Ciri-ciri Sistem Pakar

Ciri-ciri sistem pakar menurut Hayadi (2016:3), adalah:

1. Terbatas pada *domain* keahlian tertentu
2. Dapat memberikan penalaran untuk data yang tidak pasti
3. Dapat mengemukakan rangkaian alasan yang diberikannya dengan cara yang dapat dipahami
4. Berdasarkan pada kaidah atau *rule* tertentu
5. Dirancang untuk dapat dikembangkan secara bertahap
6. Pengetahuan dan mekanisme *inferensi* jelas terpisah
7. Keluarannya bersifat anjuran
8. Sistem dapat mengaktifkan kaidah secara searah yang sesuai yang dituntun oleh dialog dengan pemakai

d. **Klasifikasi Sistem Pakar**

Klasifikasi sistem pakar berdasarkan kegunaannya menurut Siswanto dalam Hayadi (2016:4) yaitu:

1. **Diagnosis :**
 - a. Rekomendasi obat untuk orang sakit, kerusakan mesin, kerusakan rangkaian elektronik, dsb.
 - b. Menemukan apa masalah atau kerusakan yang terjadi
 - c. Menggunakan pohon keputusan (*decision tree*) sebagai representasi pengetahuannya.
2. **Pengajaran**
 - a. Digunakan untuk pengajaran, mulai dari SD sampai dengan PT.
 - b. Membuat diagnosa apa penyebab kekurangan dari siswa, kemudian memberikan cara untuk memperbaikinya.
3. **Interpretasi**, untuk menganalisa data yang tidak lengkap, tidak teratur, dan data yang kontradiktif seperti interpretasi citra.
4. **Prediksi** (peramalan cuaca dan penentuan masa tanam)
5. **Perencanaan**
 - a. Perencanaan mesin-mesin sampai dengan manajemen bisnis
 - b. Efisiensi biaya, waktu, dan material
6. **Kontrol**
 - a. Kontrol kegiatan yang membutuhkan presisi waktu tinggi
 - b. Pengontrolan pada industry-industri berteknologi tinggi.

e. **Representasi Pengetahuan**

Menurut Kusriani (2008:6) menjelaskan bahwa:

Representasi pengetahuan merupakan metode yang digunakan untuk mengkodekan pengetahuan dalam sebuah sistem pakar. Representasi dimaksudkan untuk menangkap sifat-sifat penting masalah dan membuat informasi itu dapat diakses oleh prosedur pemecahan masalah.

Adapun karakteristik dari metode representasi pengetahuan menurut Kusrini (2008:6) adalah:

1. Harus bisa diprogram dengan bahasa pemrograman atau dengan *shells* dan hasilnya disimpan dalam memori.
2. Dirancang sedemikian sehingga isinya dapat digunakan untuk proses penalaran.
3. Model representasi pengetahuan merupakan sebuah struktur data yang dapat dimanipulasi oleh inferensi dan pencarian untuk aktifitas pencocokan pola.

Dalam sistem pakar ada beberapa metode representasi pengetahuan menurut Kusrini (2008:6):

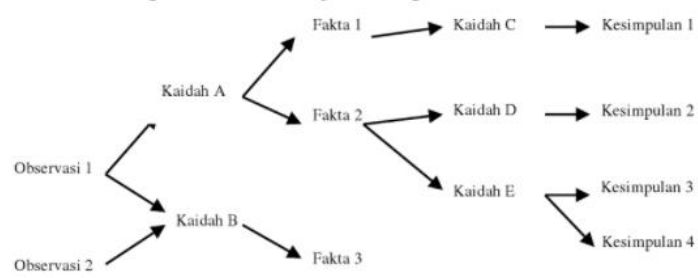
1. Metode jaringan semantik, *frame* dan logika predikat dapat digunakan untuk pengetahuan yang bersifat deklaratif
2. Kaidah Produksi, digunakan untuk pengetahuan procedural yang merepresentasikan aksi dan prosedur.

f. **Metode Inferensi**

Inferensi merupakan proses untuk menghasilkan informasi dari fakta yang diketahui atau diasumsikan. Inferensi adalah konklusi logis (*logical conclusion*) atau implikasi berdasarkan informasi informasi yang tersedia yang dilakukan dalam suatu modul yang disebut *Inference Engine* (Mesin Inferensi) (Kusrini, 2008:8).

Ada 2 metode inferensi yang penting dalam sistem pakar, menurut Hayadi (2016:7) dijelaskan sebagai berikut:

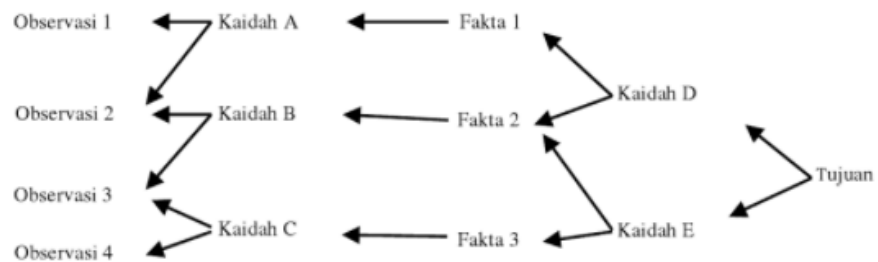
1. Runut Maju (*Forward Chaining*), apabila pelacakan kesimpulan dilakukan dengan runut kebelakang yaitu memulai dari sekumpulan data menuju kesimpulan. Langkah-langkah yang harus dilakukan dalam membuat sistem *forward chaining* berbasis aturan yaitu :
 - a) Pendefinisian masalah, meliputi pemilihan domain masalah dan akuisisi pengetahuan.
 - b) Pendefinisian data input, dimana data awal untuk diperlukan untuk memulai inferensi.
 - c) Pendefinisian struktur pengendalian data, penggunaan premis ambahan untuk membantu mengendalikan pengaktifan suatu aturan.
 - d) Penulisan kode awal, untuk menentukan apakah sistem telah menangkap domain pengetahuan secara efektif dalam struktur aturan yang baik.
 - e) Perancangan antarmuka
 - f) Pengujian sistem, untuk menguji sejauh mana sistem berjalan dengan benar.
 - g) Pengembangan sistem, meliputi penambahan antarmuka dan pengetahuan sesuai dengan *prototype* sistem.
 - h) Evaluasi sistem, dengan melakukan pengujian sistem dengan masalah yang sebenarnya.



Sumber: Hayadi (2016:7)

Gambar II.3
Diagram *Forward Chaining*

2. Runut Balik (*Backward Chaining*), apabila pelacakan dimulai dengan penalaran dari kesimpulan hipotesa menuju fakta yang mengandung hipotesa tersebut.



Sumber: Hayadi (2016:7)

Gambar II.4
Diagram *Backward Chaining*

2.1.2 Konsep Dasar Program

a. Web Server

Sebuah perangkat lunak dalam server yang berfungsi menerima permintaan (*request*) berupa halaman web melalui HTTP atau HTTPS dari klien yang dikenal dengan *browser* web dan mengirimkan kembali

(*response*) hasilnya dalam bentuk halaman-halaman web yang umumnya berbentuk dokumen HTML. (Solichin, 2016:7)

b. Server Side Scripting

Teknologi *scripting* atau pemrograman *web* dimana *script* (program) dikompilasi atau diterjemahkan di *server* sehingga memungkinkan untuk menghasilkan halaman *web* yang dinamis. Beberapa contoh *server side scripting* (Solichin, 2016:8):

- 1) ASP (Active Server Page) dan ASP.NET
- 2) ColdFusion (<http://www.macromedia.com/software/coldfusion>)
- 3) Java *Server Pages* (<http://java.sun.com/products/jsp/>)
- 4) Perl (<http://www.perl.org>)
- 5) Python (<http://www.python.org>)
- 6) PHP (<http://www.php.net>)

c. PHP

PHP yang merupakan kependekan dari PHP: *Hypertext Preprocessor* pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1994. Karakteristik PHP menurut Solichin (2016:14) adalah sebagai berikut:

- 1) Harus ditulis dalam tag standar :
 - a) `<? dan ?>`
 - b) `<? Php ?>`
 - c) `<script language="php">` dan `</script>`
 - d) `<% dan %>`

- 2) Setiap satu *statement* (perintah) diakhiri dengan titik-koma (;)
- 3) *Case sensitive* untuk nama *identifier* yang dibuat oleh *user* (berupa variabel, konstanta, fungsi, dll), namun tidak *case sensitive* untuk *identifier built-in* dari PHP. Sehingga :
 - a) \$nama ≠ \$Nama ≠ \$NAMA
 - b) echo = ECHO
- 4) Variabel, digunakan untuk menyimpan sebuah *value*, data atau informasi.
 - a) Nama variabel diawali dengan tanda \$ dengan panjang tidak terbatas, dimana setelah tanda \$ diawali oleh huruf atau *under-score* (_).
 - b) Karakter berikutnya bisa terdiri dari huruf, angka, dan karakter tertentu yang diperbolehkan.
 - c) Bersifat *case-sensitive*
 - d) Tidak perlu dideklarasikan
 - e) Tidak boleh mengandung spasi
- 5) PHP yang mendukung 8 tipe data, yaitu :
 - a) boolean
 - b) integer
 - c) float
 - d) string
 - e) array
 - f) object
 - g) resource
 - h) NULL

- 6) Konstanta, variabel konstan yang nilainya tidak berubah-ubah. Untuk mendefinisikan konstanta dalam PHP, digunakan fungsi *define()*

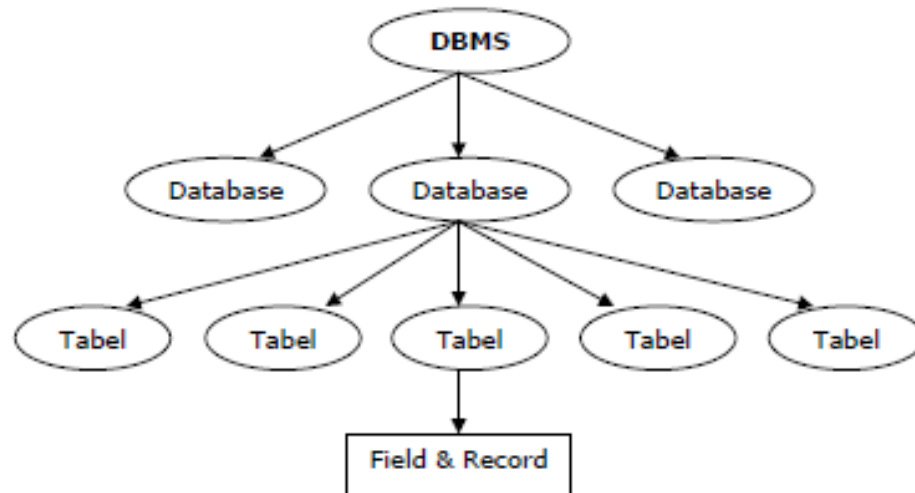
d. *Database*

Database digunakan untuk menyimpan informasi atau data yang terintegrasi dengan baik di dalam komputer. Untuk mengelola *database* diperlukan suatu perangkat lunak yang disebut DBMS (*Database Management System*). DBMS memungkinkan *user* (pengguna) untuk membuat, memelihara, mengontrol, dan mengakses *database* secara praktis dan efisien sehingga *user* akan lebih mudah mengontrol dan memanipulasi data yang ada (Solichin, 2016:84).

Beberapa *software* atau perangkat lunak DBMS yang sering digunakan dalam aplikasi program antara lain:

- a) DB2 - <http://www-306.ibm.com/software/data/db2/>
- b) Microsoft SQL Server - <http://www.microsoft.com/sql/>
- c) Oracle - <http://www.oracle.com>
- d) Sybase - <http://www.sybase.com/>
- e) Interbase - <http://www.borland.com/interbase>
- f) Teradata - <http://www.teradata.com/>
- g) Firebird - <http://www.firebirdsql.org/>
- h) MySQL – <http://www.mysql.com>
- i) PostgreSQL - <http://www.postgresql.org/>

Dalam konsep *database*, urutan atau hierarki *database* digambarkan dalam gambar II.5:



Sumber: Solichin (2016:84)

Gambar II.5
Hierarki database

e. MySQL

Beberapa kelebihan MySQL dalam pemrograman (Solichin, 2016:85), antara lain:

- 1) *Free* (bebas didownload)
- 2) Stabil dan tangguh
- 3) Fleksibel dengan berbagai pemrograman
- 4) *Security* yang baik
- 5) Dukungan dari banyak komunitas
- 6) Kemudahan management database.
- 7) Mendukung transaksi
- 8) Perkembangan *software* yang cukup cepat.

Berikut ini beberapa *tools* yang biasa digunakan dalam mempermudah administrasi *server* MySQL (Solichin, 2016:96):

- 1) MySQL *Command Line Client*
- 2) My-SQL *Front*
- 3) PHPMyAdmin
- 4) SQLYog
- 5) MySQL *Administrator* dan MySQL *Query Browser*

f. SQL

SQL merupakan singkatan dari *Structured Query Language*. SQL atau juga sering disebut sebagai *query* merupakan suatu bahasa (*language*) yang digunakan untuk mengakses *database*. SQL dikenalkan pertama kali dalam IBM pada tahun 1970 dan sebuah standar ISO dan ANSI ditetapkan untuk SQL. Terdapat 2 (dua) jenis perintah SQL, yaitu (Solichin, 2016:99):

1) DDL atau *Data Definition Language*

DDL merupakan perintah SQL yang berhubungan dengan pendefinisian suatu struktur database, dalam hal ini *database* dan tabel. Beberapa perintah dasar yang termasuk DDL ini antara lain: CREATE, ALTER, RENAME, DROP

2) DML atau *Data Manipulation Language*

DML merupakan perintah SQL yang berhubungan dengan manipulasi atau pengolahan data atau *record* dalam *table*. Perintah SQL yang termasuk dalam DML antara lain: SELECT, INSERT, UPDATE, DELETE

g. Hosting, HTTP, FTP, dan Domain

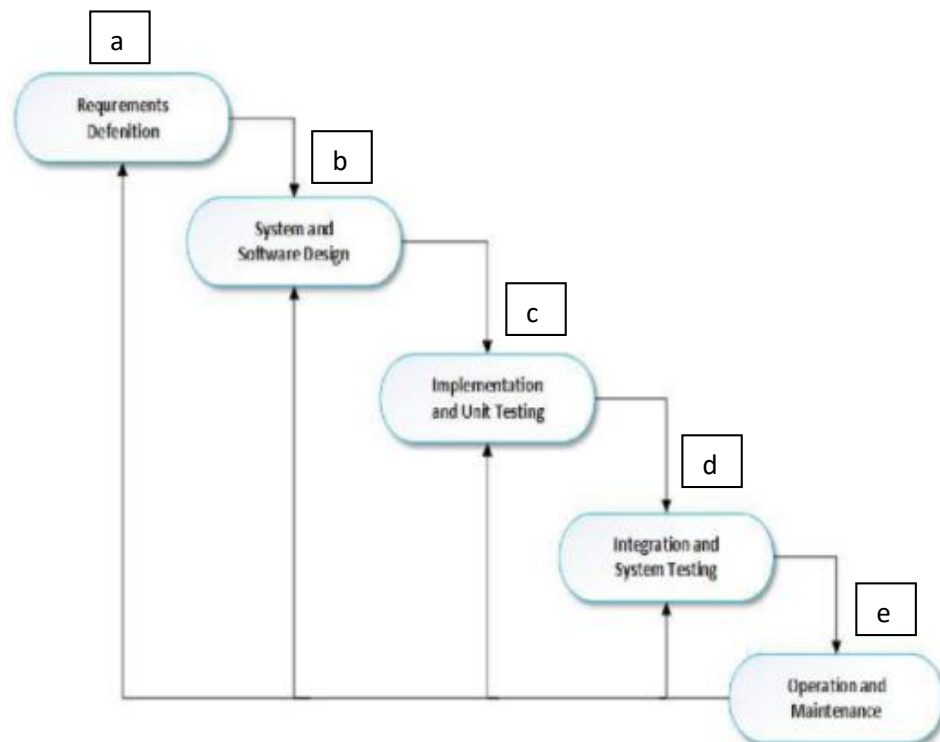
Menurut Meissa (2009:190), ada beberapa hal yang perlu dilakukan sebelum melakukan publikasi atas sebuah *website* diantaranya:

- 1) Hosting, dianalogikan sebagai rumah untuk *website* yang berfungsi untuk menyimpan semua data-data *website* yang terhubung dengan internet selama 24 jam sehingga *website* dapat dikunjungi kapan saja.
- 2) HTTP, yaitu sebuah protocol yang memungkinkan pengguna internet mengunjungi sebuah *website* yang ada di *server* dengan hanya memasukkan alamat ke *browser*. Contoh : <http://namadomain.com>
- 3) FTP, yaitu sebuah protokol yang memungkinkan pemilik website melakukan upload *file-file* ke dalam hosting melalui sebuah aplikasi FTP.
- 4) Domain, adalah sebuah nama atau alamat dari website yang dimiliki seseorang di internet. Selain dengan nama seperti www.domain.com, domain juga bisa berbeda-beda ekstensinya seperti .net, .org, .info, .name, dan lain-lain.

2.1.3 Metode Pengembangan Sistem *Waterfall*

Model *Waterfall* atau disebut juga dengan “*classic life cycle*” ini muncul pertama kali sekitar tahun 1970 sehingga sering dianggap kuno. Akan tetapi, metode inilah yang banyak digunakan di dalam *Software Engineering* (SE). Pada metode ini, tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan (Muharto dan Ambarita, 2016:105).

Menurut Pressman dan Sommerville dalam Muharto dan Ambarita (2016:105), metode pengembangan sistem model *waterfall* terdiri dari beberapa langkah, seperti ditunjukkan pada gambar II.2 berikut:



Sumber: Muharto dan Ambarita (2016:105)

Gambar II.6 Metode Waterfall

Penjelasan dari alur diatas sebagai berikut:

- a. Analisis dan definisi kebutuhan. Layanan, batasan, dan tujuan sistem ditentukan melalui konsultasi dengan *user* atau pemakai.
- b. Perancangan sistem dan perangkat lunak yang dibagi dalam sistem perangkat keras dan sistem perangkat lunak. Tahap ini menentukan arsitektur sistem secara keseluruhan. Perancangan melibatkan identifikasi dan deskripsi abstraksi sistem perangkat lunak yang mendasar.
- c. Implementasi dan pengujian unit. Pada tahap ini, perancangan perangkat lunak direalisasikan dengan program atau unit program. Pengujian ini melibatkan verifikasi bahwa setiap unit telah memenuhi spesifikasinya.

- d. Integrasi dan pengujian sistem. Unit program atau program individual diintegrasikan diintegrasikan dan diuji sebagai sistem yang lengkap untuk menjamin bahwa kebutuhan sistem telah dipenuhi.
- e. Operasi dan pemeliharaan, yaitu mengoperasikan program di lingkungannya dan melakukan pemeliharaan. Biasanya ini merupakan fase siklus hidup yang paling lama. Pemeliharaan mencakup koreksi dari berbagai *error* yang tidak ditemukan pada tahap-tahap sebelumnya, melakukan perbaikan atas implemenasi unit sistem dan pengembangan layanan sistem, dan persyaratan-persyaratan baru ditambahkan.

2.1.4 Perangkat Pendukung

a. *Unified Modeling Language (UML)*

UML adalah sekumpulan alat yang digunakan untuk melakukan abstraksi terhadap sebuah sistem atau perangkat lunak berbasis objek sehingga mempermudah pengembangan aplikasi yang berkelanjutan. Aplikasi atau sistem yang tidak terdokumentasi biasanya dapat menghambat pengembangan karena *developer* harus melakukan penelusuran dan mempelajari kode program. UML menjadi alat bantu untuk transfer ilmu tentang sistem atau aplikasi yang akan dikembangkan dari satu *developer* ke *developer* lainnya.

Dalam pengembangan berorientasi objek ada beberapa prinsip yang harus dikenal (Fajar, 2016):

- 1) Object
- 2) Class
- 3) Abstraction

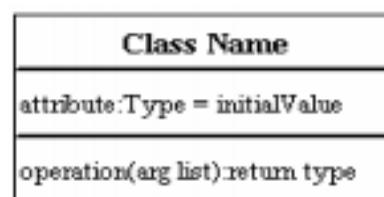
- 4) Encapsulation
- 5) Inheritance
- 6) Polymorphism

Dalam UML sendiri terdapat beberapa diagram yang wajib dikuasai yaitu (Fajar, 2016):

1) Struktural Diagram

a) *Class* Diagram, diagram ini terdiri dari *class*, *interface*, *association*, dan *collaboration*. Diagram ini menggambarkan objek - objek yang ada di sistem. Simbol dan notasi dasar *Class* Diagram menurut Syafei (2016:16), dijelaskan sebagai berikut :

- (1) *Class*, digambarkan dengan bentuk empat persegi panjang yang dibagi kedalam ruang-ruang terpisah (*compartments*). Nama *class* ditempatkan pada bagian pertama (rata tengah, di boldm dan Huruf besar), daftar atribut diletakan pada bagian kedua, dan operasi-operasi pada *class* dibagian ketiga.



Sumber: Syafei (2016:16)

Gambar II.7
Notasi Class

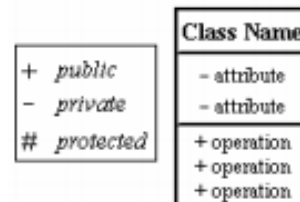
- (2) *Active Classes*, *class* yang memulai dan mengontrol aliran atau arus atau arah aktifitas. Sementara *passive class* menyimpan data dan melayani *class* yang lain.



Sumber: Syafei (2016:16)

Gambar II.8
Notasi Active Class

- (3) *Visibility*, untuk menandakan siapa yang bisa mengakses informasi-informasi yang diisi kedalam sebuah *class*. *Private visibility* artinya informasi yang ada dalam sebuah *class* disembunyikan atau dipartisi dari pihak luar. *Public visibility* mengizinkan semua *class* yang lainnya untuk melihat nilai informasi. *Protected visibility* mengizinkan *class-class* yang ada yang merupakan turunannya untuk mengakses informasi yang ada didalamnya karena mereka merupakan *class* turunan dari *class* induknya (*inherited*).

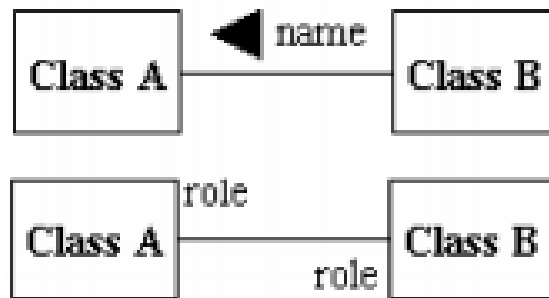


Sumber: Syafei (2016:17)

Gambar II.9
Notasi Visibility

- (4) *Associations*, representasi atau gambaran relasi status diantara *class-class*. Penempatan nama *associations* ada pada bagian atas atau dibawah garis *associations*, dengan menggunakan tanda anak panah yang berisi sebuah kata yang mengindikasikan relasi secara langsung. Peletakan *rule* (aturan

atau ketentuan) pada bagian akhir *associations*. Aturan tersebut merepresentasikan arah bagi kedua kelas untuk saling berhubungan satu sama lain.




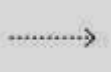








Sumber: Syafei (2016:17)

Gambar II.10
Notasi Association

- b) *Object Diagram*, diagram ini menggambarkan hasil instansi dari *class diagram*. Diagram ini digunakan untuk membuat *prototype*.
 - c) *Component Diagram*, diagram ini menggambarkan kumpulan komponen dan hubungan antar komponen. Komponen terdiri dari *class*, *interface*, atau *collaboration*
 - d) *Deployment Diagram*, diagram ini menggambarkan kumpulan *node* dan hubungan antar *node*. *Node* adalah entitas fisik dimana komponen di-*deploy*. Entitas fisik ini dapat berupa *server* atau perangkat keras lainnya.
- 2) *Behavioral Diagram*
- a) *Use case Diagram*, diagram ini menggambarkan kumpulan *use case*, aktor, dan hubungan mereka. *Use case* adalah hubungan antara fungsionalitas sistem dengan aktor internaleksternal dari sistem.

Adapun notasi-notasi yang ada pada *use case diagram* dapat dilihat pada tabel II.1 berikut ini:

Tabel II.1
Notasi Pada Use Case Diagram

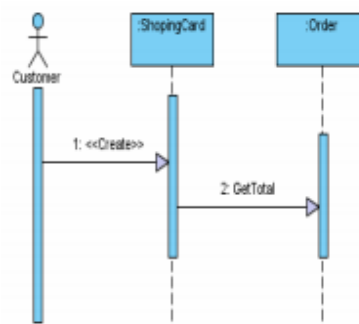
NO	GAMBAR	NAMA	KETERANGAN
1		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
2		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (<i>independent</i>).
3		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>).
4		<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara <i>eksplisit</i> .
5		<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
6		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
9		<i>Collaboration</i>	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (<i>sinergi</i>).
10		<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

Sumber: <http://www.kajianpustaka.com/2013/12/use-case-diagram.html>

b) *Sequence Diagram*, diagram ini menggambarkan interaksi yang menjelaskan bagaimana pesan mengalir dari objek ke objek

lainnya. Simbol dan notasi dasar *sequence diagram* menurut Syafei, (2016:33) adalah sebagai berikut :

- (1) *Class Roles*, menggambarkan way (jalan) sebuah objek akan menunjukkan reaksi (berkelakuan) dalam sebuah keadaan (konteks).
- (2) *Activation boxes*, merepresentasikan waktu yang dibutuhkan oleh sebuah objek untuk melaksanakan sebuah tugas atau perintah secara lengkap.
- (3) *Messages*, anak panah yang merepresentasikan komunikasi antara objek yang berguna untuk mengirimkan perintah kepada *LifeLine*. Message bisa berasal dari *actor* kepada *LifeLine* atau dari *LifeLine* kepada *LifeLine* yang lain. Untuk merepresentasikan pesan-pesan *asynchronous* digunakan setengah garis anak panah. Pesan ini dikirim dari sebuah objek yang tidak akan menunggu respon dari penerima sebelum melanjutkan perintahnya.








Sumber: Syafei (2016:34)

Gambar II.11
Contoh Implementasi Messages

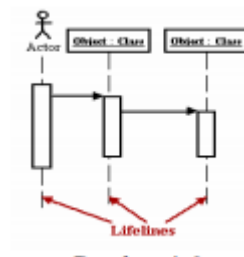
Berbagai tipe message bisa dilihat pada tabel berikut:

Tabel II.2
Tabel Tipe *Message*

Arrow	Message type
	Simple
	Synchronous
	Asynchronous
	Balking
	Time out

Sumber: Syafei (2016:34)

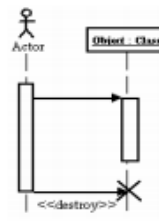
- (1) *LifeLines*, tanda garis pisah yang mengindikasikan kehadiran objek pada saat terakhir.



Sumber: Syafei (2016: 34)

Gambar II.12
LifeLines

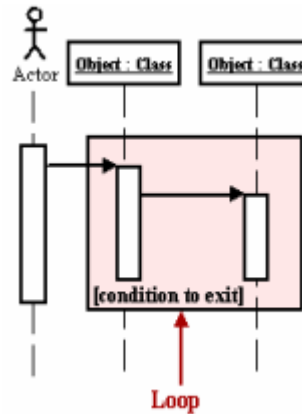
- (2) *Destroying Objects*, Ojek dapat diakhiri secara lebih cepat menggunakan sebuah garis anak panah yang diberi label “<<destroy>>” dan diujungnya diberi label X.



Sumber: Syafei (2016:34)

Gambar II.13
Destroying Object

- (3) *Loops*, suatu pengulangan yang digambarkan seperti sebuah empat persegi panjang, dengan kondisi untuk keluar pada pengulangan di bagian kiri bawah di dalam kurang empat persegi panjang.

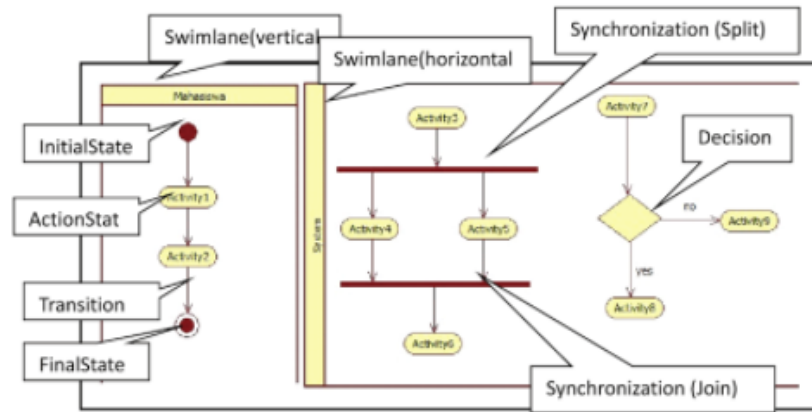


Sumber: Syafei (2016:34)

Gambar II.14
Loop

- c) *Collaboration Diagram*, diagram ini merupakan bentuk lain dari *sequence diagram*. Diagram ini menggambarkan struktur organisasi dari sistem dengan pesan yang diterima dan dikirim.
- d) *Statechart Diagram*, diagram ini menggambarkan bagaimana sistem dapat bereaksi terhadap suatu kejadian dari dalam atau luar. Kejadian (*event*) ini bertanggung jawab terhadap perubahan keadaan sistem.
- e) *Activity Diagram*, menggambarkan aliran kontrol sistem. Diagram ini digunakan untuk melihat bagaimana sistem bekerja ketika dieksekusi. Notasi yang umum digunakan dalam *Activity*

Diagram (Triandini dan Suardika, 2012:37) tercantum pada gambar II.15 berikut :



Sumber: (Triandini dan Suardika, 2012:37)

Gambar II.15
Notasi Activity Diagram




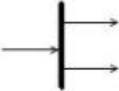
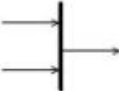

Penjelasan atas masing-masing notasi tersebut sebagai berikut:

- 1) *Swimlane*, mewakili agen yang melakukan aktivitas. Dalam alur kerja, umumnya mempunyai agen yang berbeda yang melakukan langkah yang berbeda dari proses alur kerja. Simbol *swimlane* membagi aktivitas alur kerja ke dalam kelompok yang menunjukkan agen mana yang menjalankan aktivitas yang mana. Ada dua jenis *swimlane* yang dapat digunakan sesuai dengan kebutuhan yaitu *swimlane vertical* dan *swimlane horizontal*.
- 2) *InitialState*, awal dari alur kerja
- 3) *ActionState*, melambangkan aktivitas tersendiri dalam alur kerja.
- 4) *Transition*, melambangkan aktivitas tersendiri dalam alur kerja.

- 5) *FinalState*, akhir dari alur kerja
- 6) *Synchronization*, membagi alur kerja menjadi beberapa alur yang bersamaan ataupun menggabungkan lagi alur yang bersamaan.
- 7) *Decision*, titik pengambilan keputusan dimana aliran proses tersebut akan mengikuti satu jalur atau jalur lainnya.

Adapun elemen-elemen pada *Activity Diagram*, tercantum pada tabel II.3 berikut ini:

Tabel II.3
Elemen Activity Diagram

Simbol	Keterangan
	Start Point
	End Point
	Activities
	Fork (Percabangan)
	Join (Penggabungan)
	Decision
Swimlane	Sebuah cara untuk mengelompokkan activity berdasarkan Actor (mengelompokkan activity dalam sebuah urutan yang sama)

Sumber: <https://www.codepolitan.com/mengenal-uml-contoh-uml-diagram-model-activity-diagram>

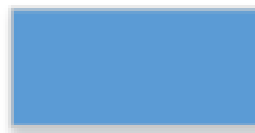
b. *Entity Relation Diagram (ERD)*

Menurut Yanto (2016:32) menjelaskan bahwa:

ERD adalah suatu diagram untuk menggambarkan desain konseptual dari model konseptual suatu basis data relasional. ERD juga merupakan gambaran yang merelasikan antara objek yang satu dengan objek yang lain dari objek di dunia nyata yang sering dikenal dengan hubungan antar entitas.

ERD terdiri atas 3 notasi dasar menurut Kusri (2016:21), yaitu:

- 1) Entitas (*Entity*), dilambangkan dengan bentuk persegi panjang, seperti tampak pada gambar berikut :



Sumber: Kusri (2007:21)

Gambar II.16
Notasi Entitas

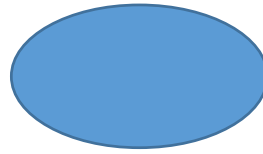
- 2) Relasi (*Relationship*), hubungan diantara beberapa *entity*. *Relationship Set* adalah sekumpulan relasi yang mempunyai tipe yang sama. *Relationship set* digambarkan dengan *diamond* seperti tampak pada gambar berikut :



Sumber: Kusri (2007:21)

Gambar II.17
Notasi Relasi

- 3) Atribut (*attribute*), sebutan untuk mewakili suatu *entity*. Atribut dalam ERD dilambangkan dengan bentuk *elips*, seperti gambar dibawah :



Sumber: Kusrini (2007:22)

Gambar II.18
Notasi Atribut

Suatu *field* yang dapat mewakili suatu record disebut dengan istilah *Key* atau Kunci (Yuhefizard, 2008:23). Ada beberapa jenis *Key* (Kunci) yang perlu kita ketahui menurut Yanto (2016:37), diantaranya:

- 1) *Super Key*, satu atribut atau kumpulan atribut yang secara unik mengidentifikasi sebuah baris di dalam relasi atau himpunan dari satu atau lebih entitas yang dapat digunakan untuk mengidentifikasi secara unik sebuah entitas dalam set entitas.
- 2) *Candidate Key*, atribut yang menjadi determinan yang dapat dijadikan identitas dari sebuah relasi.
- 3) *Primary Key*, kandidat *key* yang dipilih untuk mengidentifikasi baris data secara unik dalam relasi.
- 4) *Alternative Key*, kandidat *key* yang tidak terpilih sebagai *primary key* atau atribut untuk menggantikan kunci utama.
- 5) *Foreign Key*, atribut dengan domain yang sama menjadi kunci utama sebuah relasi, tetapi pada relasi lain atribut tersebut berperan sebagai atribut biasa.
- 6) *Composite Key*, kunci yang terdiri dari dua atribut atau lebih. Atribut tersebut jika berdiri sendiri tidak menjadi identitas baris, tetapi bila

dirangkaikan menjadi satu kesatuan akan dapat mengidentifikasi secara unik.

Menurut Whitten et.al dalam Lubis (2016:41), mendefinisikan bahwa kardinalitas adalah:

Mendefinisikan batas minimal dan maksimal tiap kejadian atau peristiwa pada satu entitas yang mungkin direlasikan untuk satu kejadian pada entitas yang lain. Karena semua hubungan secara langsung, kardinal harus dapat didefinisikan secara langsung diantara kedua entitas setiap dihubungkan.

Menurut Yuhefizard (2008:15) yang mengemukakan bahwa hubungan antar tabel dalam ERD, terdiri atas 4 jenis yaitu:

- 1) Satu ke sat (*One to One*), apabila satu record pada *entity* A ber-relasi paling banyak satu record juga pada *entity* B, begitu juga sebaliknya, satu *record* pada *entity* B, ber-relasi paling banyak satu *record* juga dengan *entity* A. Dalam ERD, relasi ini disimbolkan dengan angka 1.
- 2) Satu ke Banyak (*One to Many*), apabila satu *record* pada *entity* A ber-relasi dengan beberapa *record* pada *entity* B, tapi tidak sebaliknya, setiap *record* pada *entity* B ber-relasi paling banyak satu *record* dengan *entity* A. Dalam ERD, relasi ini disimbolkan dengan angka 1 (*one*) untuk menyatakan satu dan huruf M atau N untuk menyatakan banyak (*many*).
- 3) Banyak ke Satu (*Many to One*), ini adalah kebalikan dari relasi *one to many* dimana setiap *record* pada *entity* A hanya dapat ber-relasi paling banyak 1 *record* pada *entity* B, tapi tidak sebaliknya, satu *record* pada *entity* B dapat ber-relasi dengan beberapa *record* pada

entity A. Dalam ERD, relasi ini disimbolkan dengan angka 1 untuk menyatakan satu dan huruf M atau N untuk menyatakan banyak.

- 4) Banyak ke Banyak (*Many to Many*), apabila ada beberapa *record* pada *entity* A dapat ber-relasi dengan beberapa *record* juga pada *entity* B, begitu juga sebaliknya, beberapa *record* pada *entity* B dapat ber-relasi dengan beberapa *record* juga pada *entity* A. Dalam ERD, relasi ini disimbolkan dengan huruf M atau N untuk menyatakan banyak.

c. Adobe Dreamweaver CS6

Menurut Hikmah (2015:1) memberikan penjelasan mengenai Adobe Dreamweaver bahwa:

Adobe Dreamweaver merupakan salah satu tool untuk mendesain halaman website yang merupakan tools keluaran Adobe yang dahulu merupakan macromedia. Adobe Dreamweaver merupakan salah satu tool yang banyak digunakan oleh pengembang web dikarenakan halamannya berbasis GUI (Grafical User Interface) sehingga memudahkan pengembang untuk mengembangkan website tidak terpaku konsep tekstual.

2.1.5 Pengujian Sistem (*Testing*)

Ada 2 jenis pengujian sistem menurut Maturidi (2014:68), yaitu:

a. *Black Box Testing*

Test case ini bertujuan untuk menunjukkan fungsi PL tentang cara beroperasinya, apakah pemasukan data keluaran telah berjalan sebagaimana yang diharapkan dan apakah informasi yang disimpan secara eksternal selalu dijaga kemutakhirannya.

Pengujian ini berfokus pada persyaratan fungsional perangkat lunak sehingga memungkinkan analis sistem memperoleh kumpulan kondisi input

yang akan mengerjakan seluruh keperluan fungsional program. Pengujian ini tidak terfokus pada struktur kontrol seperti pada pengujian *whitebox* tetapi pada domain informasi. Adapun tujuan metode ini adalah mencari kesalahan pada:

1. Fungsi yang salah atau hilang
2. Kesalahan pada interface
3. Kesalahan pada struktur data atau akses database
4. Kesalahan informasi
5. Kesalahan

b. White Box Testing

White Box Testing adalah meramalkan cara kerja perangkat lunak secara rinci karenanya *logical path* (jalur logika) perangkat lunak akan dites dengan menyediakan *test case* yang akan mengerjakan kumpulan kondisi dan atau pengulangan secara spesifik. Secara sekilas dapat diambil kesimpulan *white box testing* merupakan petunjuk untuk mendapatkan program yang benar secara 100%.

Uji coba *white box* adalah metode perancangan *test case* yang menggunakan struktur kontrol dari perancangan prosedural untuk mendapatkan *test case*. Melalui pengujian ini, analis sistem akan dapat memperoleh *test case* yang menjamin hal-hal sebagai berikut:

1. Seluruh *independent path* di dalam modul yang dikerjakan sekurang-kurangnya sekali
2. Seluruh keputusan logical
3. Seluruh loop yang sesuai dengan batasannya

4. Seluruh struktur data internal yang menjamin validitas

2.1.6 Psikotes

Psikotes adalah tes yang bertujuan untuk memberikan gambaran mengenai aspek-aspek psikis seseorang sesuai dengan kebutuhan, dimana tes tersebut digunakan sebagai alat atau sarana bagi Psikolog atau *Tester* untuk bisa memahami secara utuh aspek-aspek psikologis dari seseorang agar bisa memberikan gambaran (profil psikogram) setiap individu yang mengikuti tes tersebut. Tes ini dapat berbentuk tertulis, proyektif, atau evaluasi secara verbal yang teradministrasi untuk mengukur kemampuan kognitif dan emosional seseorang. (Tim Psikologi Salemba, 2016:ix).

Menurut Tim Psikologi (2014:3) mengemukakan bahwa, “Secara konseptual, psikotes bertujuan mengukur seberapa jauh kemampuan psikologis seseorang, baik menyangkut potensi diri, kemampuan khusus, bakat, minat, maupun kondisi kejiwaan. “

Psikotes terdiri dari beragam subtes yang memiliki karakter yang berbeda-beda menurut Tim Psikologi (2014:3), yaitu:

1. Subtes untuk menggali kemampuan kompetensi, seperti ketelitian, daya tahan, dan inteligensi. Untuk jenis ini, sifat penilaiannya adalah benar dan salah.
2. Subtes untuk menggali profil kepribadian seperti pengendalian emosi, minat, bakat dan tipe kepribadian. Tes ini berhubungan dengan satu keharmonisan atas hal yang akan dilakukan seseorang dalam situasi tertentu dan menghasilkan daya penyesuaian yang efektif. Pada subtes jenis ini, tidak ada

jawaban yang salah. Semua jawaban benar, karena yang digali oleh tes jenis ini adalah gambaran kepribadian diri.

a. Tes Temperamen Manusia

Menurut Allport dalam Sunaryo (2004:117) menjelaskan mengenai pengertian temperamen manusia bahwa:

Temperamen adalah gejala karakteristik dari sifat emosi individu, termasuk mudah tidaknya terkena rangsangan emosi, kekuatan dan kecepatannya bereaksi, kualitas kekuatan suasana hati secara fluktuasi dan intensitas suasana hati, serta bergantung pada faktor konstitusional, yang karenanya terutama berasal dari keturunan. Jadi, temperamen sifatnya turun-temurun dan tak dapat diubah oleh pengaruh-pengaruh dari luar.

Menurut Hipocrates dalam Dariyo (2008:109), temperamen dibagi menjadi 4 macam, yaitu:

1. Sanguinis

Antonius (2013:34) menjelaskan mengenai beberapa karakteristik tipe temperamen ini:

- a) Orang yang sangat bersemangat dalam hidup. Selalu tampak ceria, hangat, bersahabat, dan sangat menikmati hidup.
- b) Memiliki sifat yang mudah menerima sehingga kesan-kesan dari luar dapat dengan mudah masuk ke dalam hati.
- c) Cenderung lebih mendasarkan perasaannya daripada pemikirannya saat mengambil keputusan.
- d) Sangat menyukai 'kesenangan', dan jarang sekali membiarkan hatinya bersedih berlama-lama.
- e) Tipe orang yang sangat suka bicara, dengan mudah mereka dapat menularkan perasaan 'semangat' kepada orang lain melalui perkataan.

- f) Gaya yang gaduh, bersuara keras, dan ramah membuat orang sanguinis tampak percaya diri lebih daripada yang sebenarnya.

2. Melankolis

@PsikologID (2013:36) menjelaskan, mengenai beberapa karakteristik tipe temperamen ini:

- a) Memiliki rasa seni yang tinggi, kemampuan analitis yang kuat, perfeksionis, sensitif, berbakat, dan rela berkorban.
- b) Perasaan sangat berpengaruh pada pribadi melankolis cenderung introvert, tapi apabila dia sedang berada dalam puncak sukacitanya, bisa saja menjadi lebih ekstrovert.
- c) Pribadi yang sangat perfeksionis cenderung membuat mereka sering menyalahkan diri sendiri dan menjadi rendah diri, padahal mereka memiliki daya analitik yang hebat yang mampu memperhitungkan secara akurat bahaya dan halangan dalam setiap proyek.
- d) Cenderung memilih pekerjaan yang membutuhkan pengorbanan dan ketekunan, sekali memilih sesuatu maka mereka akan tetap setia mengerjakannya.

3. Plegmatis

Antonius (2013:37) menjelaskan, mengenai beberapa karakteristik tipe temperamen ini:

- a) Memiliki sifat alamiah pendamai dan menghindari kekerasan.
- b) Orang yang mudah diajak bergaul, ramah, dan menyenangkan.

- c) Orang yang bisa membuat sekelompok orang tertawa terbahak-bahak oleh humor-humor keringnya, tetapi mereka sendiri tidak tertawa.
- d) Pribadi yang konsisten, tenang, dan jarang sekali terpengaruh dengan lingkungan. Inilah yang membuat mereka hamper-hampir tidak pernah terlihat gelisah.
- e) Dibali pribadi yang dingin dan malu-malu, sesungguhnya mereka memiliki kemampuan untuk dapat lebih merasakan emosi yang terkandung pada sesuatu.
- f) Karena menyukai kedamaian dan tidak menyukai pertikaian, orang phlegmatis cenderung menarik diri dari segala macam keterlibatan.

4. Koleris

@PsikologID (2013:35) menjelaskan, mengenai beberapa karakteristik tipe temperamen ini:

- a) Memiliki kemauan keras dalam mencapai sesuatu.
- b) Pribadi yang aktif, praktis, cekatan, mandiri, dan sangat independen.
- c) Cenderung bersikap tegas dan berpendirian keras dalam mengambil keputusan bagi diri sendiri dan bagi orang lain.
- d) Sangat menyukai aktivitas, sehingga tidak perlu dirangsang oleh lingkungannya, tetapi justru merangsang lingkungannya melalui ide-ide yang tidak pernah berakhir, rancangan, sasaran, dan ambisi.
- e) Bukan tipe yang mudah menyerah terhadap tekanan dari orang lain. Bahkan tekanan tersebut justru semakin mendorong mereka untuk terus maju.

- f) Bagian yang sedikit berkembang dari seorang koleris adalah emosinya. Tidak mudah bersimpati kepada orang lain.
- g) Bukan orang yang mudah mengekspresikan perasaan kepada orang lain dan cenderung tidak peka terhadap kebutuhan orang lain.
- h) Cenderung dominan dan memiliki jiwa kepemimpinan yang kuat.

b. Dominasi Gaya Belajar

Dikutip dari *Intel® Teach Program - Assessing Projects* (2007:1), “Cara yang paling sederhana dan umum dari pengidentifikasian berbagai gaya belajar yang berbeda didasarkan pada indera. Umumnya disebut model VAK, kerangka kerja ini menjelaskan pelajar sebagai melihat, mendengar atau bergerak.”

@PsikologID (2013:6) menjelaskan lebih lanjut mengenai tipe gaya belajar tersebut:

1. Tipe Visual, adalah tipe orang yang cenderung menerima informasi paling banyak dan paling efektif menggunakan indera penglihatan (visual).
2. Tipe *Auditory*, adalah tipe orang yang cenderung menerima informasi paling banyak dan paling efektif menggunakan indera pendengaran (Audio).
3. Tipe Kinestetik, adalah tipe orang yang cenderung menerima informasi paling banyak dan paling efektif dengan melibatkan gerakan tubuh, peragaan, dan aktivitas fisik.

Adapun contoh media pembelajaran yang dapat digunakan untuk masing-masing tipe gaya belajar tersebut tercantum pada tabel II.4 berikut ini:

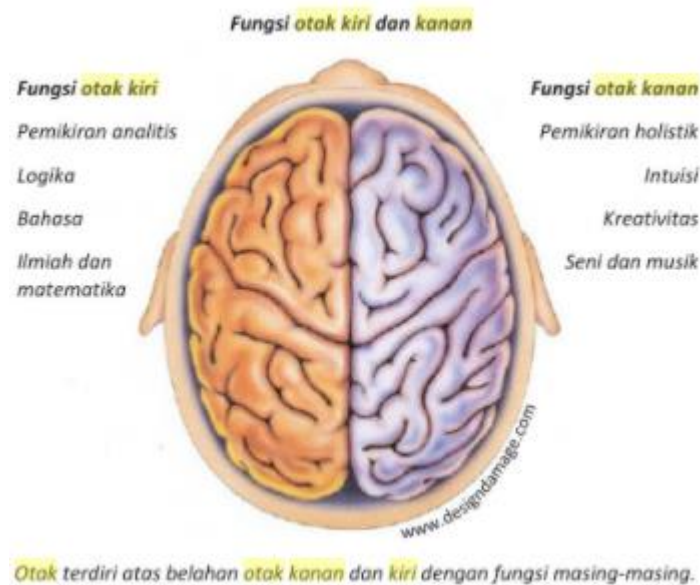
Tabel II.4
Gaya Belajar V-A-K

Penglihatan (Visual)	Gambar, video, grafik, diagram, tabel, model
Pendengaran (<i>Audiotory</i>)	Mengajar, merekam, bercerita, musik, penjelasan verbal, tanya jawab
Gerakan (Kinestetik)	Bertindak, bermain peran, membuat bentuk dengan tanah liat

Sumber: Intel® Teach Program - *Assessing Projects* (2007:1)

c. Dominasi Otak Kiri atau Otak Kanan

Otak kiri dan kanan masing-masing memiliki fungsi yang berbeda. Namun demikian, kedua belahan otak ini sama penting dan sama kuatnya, dimana mereka saling melengkapi satu dengan yang lain. Ketika seseorang mampu menggunakan kedua belahan otak tersebut dengan seimbang, maka orang tersebut bisa dikatakan sebagai seseorang dengan pribadi yang memiliki intelegensi, *analytical thinking*, dan emosi yang stabil. (Andhika (2010:12).



Sumber: Andhika (2010:12)

Gambar II.19

Fungsi otak kiri dan kanan

Adapun karakteristik dari individu dengan dominasi otak kiri dan kanan tersebut, dijelaskan @PsikologID (2013:12) sebagai berikut:

1. Dominan Otak Kiri
 - a) Memiliki pola pikir yang banyak berhubungan dengan hal-hal kuantitatif, seperti: berpikir teoritis, intelektual, logis, linier, dan rasional.
 - b) Identik dengan kecerdasan analitis
 - c) Memiliki cara kerja sangat rapi, terstruktur, dan sistematis.
 - d) Mampu memahami hal-hal yang kompleks dan perlu pemikiran yang mendetil.
 - e) Profesi yang biasanya lebih mengandalkan otak kiri adalah seorang peneliti atau ilmuwan

f) Memiliki daya ingat singkat memuat ingatan yang logis.

2. Dominan Otak Kanan

a) Memiliki cara kerja tidak terstruktur dan cenderung tidak memikirkan hal-hal yang terlalu mendetail.

b) Profesi yang biasanya lebih mengandalkan otak kanannya adalah seniman.

c) Memiliki kemampuan memuat ingatan secara *image*. Artinya, daya ingat otak lebih lama dibandingkan otak kiri.

d. Kepribadian Manusia

Mengenai kepribadian manusia, secara singkat Sunaryo (2004:116) menyebutkan bahwa, “Kepribadian adalah kualitas seseorang yang menyebabkan ia disenangi atau tidak disenangi oleh orang lain”.

C.G. Jung dalam Sunaryo (2004:119) membedakan tipologi atau tipe-tipe kepribadian manusia menjadi 3, yaitu:

1. Introvert

Tipe kepribadian individu yang minatnya lebih mengarah ke dalam pikiran dan pengalaman sendiri. Pada saat mengalami ketegangan tenggelam menyendiri ke dalam diri sendiri dan merasa mampu mencukupi diri sendiri, atau dengan kata lain tindakannya lebih dipengaruhi oleh dunia dari dalam diri sendiri.

Introvert bersifat tertutup, suka memikirkan diri sendiri, tidak terpengaruh pujian, banyak fantasi, tidak tahan kritik, mudah tersinggung, menahan ekspresi emosinya, sukar bergaul, sukar dimengerti orang lain, suka

membesarkan kesalahannya, serta analisis dan kritik diri sendiri menjadi buah pikirannya.

@PsikologID (2013:45) menjelaskan, mengenai beberapa karakteristik tipe kepribadian ini:

- a) Keramaian akan membuat tenaga cepat hilang. Oleh karena itu, biasanya seorang introvert hanya sesekali berinteraksi, kemudian diam.
- b) Ketika sedang stress, seorang introvert lebih senang menyendiri atau hanya mau berbagi kepada satu atau dua orang yang dipercaya.
- c) Suasana sepi adalah suasana yang nyaman bagi seorang introvert. Kondisi ini seolah bisa ‘mengisi’ energi seorang introvert.
- d) Seorang introvert hanya bicara seperlunya dan hanya berbicara mengenai apa yang ingin mereka bicarakan.
- e) Pada kadar yang tinggi, orang introvert jika ditanya akan diam memikirkan apa yang mereka ucapkan, setelah itu baru berbicara.
- f) Orang introvert cenderung lebih sulit melakukan pendekatan (*approach*), tetapi dalam hal *dating* lebih unggul karena mereka biasanya bisa membuat suatu hubungan *personal* yang lebih dalam dan serius. Disinilah keunggulan orang introvert.
- g) Ketika berinteraksi dengan orang introvert, arah pembicaraan akan lebih dalam, berbeda dengan orang ekstrovert yang lebih general.
- h) Orang introvert sering disibukkan dengan dirinya sendiri dan kurang peka dengan lingkungannya. Pada akhirnya lingkungannya juga tidak dapat menerima orang introvert dengan baik. Seorang introvert tahu apa yang diinginkan tetapi sulit untuk mengkomunikasikannya kepada

orang lain. Hal ini membuat orang introvert seringkali dicap sebagai orang aneh.

- i) Untuk mengerti pemikiran orang introvert, diperlukan waktu lebih banyak dan berkomunikasi secara intens. Inilah mengapa ketika berbicara dengan seorang introvert kita akan diajak menuju pembicaraan yang lebih dalam.

2. Ekstravert (Ekstrovert)

Tipe kepribadian yang tindakannya dipengaruhi dunia luar. Ekstrovert bersifat terbuka, lincah dalam pergaulan, riang, ramah, mudah berhubungan dengan orang lain, melihat realitas dan keharusan, kebal terhadap kritik ekspresi emosinya spontan, tidak begitu merasakan kegagalan, serta tidak banyak mengadakan analisis dan kritik diri sendiri.

Selanjutnya, @PsikologID (2013:44) juga menjelaskan, mengenai beberapa karakteristik tipe kepribadian ini:

- a) Lebih senang berada di tengah keramaian. Energinya terkumpul ketika berbicara dan berinteraksi dengan banyak orang. Ketika sedang berada di keramaian, orang ekstrovert seolah-olah sedang mengisi tenaganya (*charging*). Oleh karena itu, jika orang ekstrovert sedang stress, dia akan cenderung memilih untuk berinteraksi dengan banyak temannya, entah pergi ke mall, nonton, atau sekedar jalan-jalan.
- b) Orang ekstrovert tidak nyaman dengan suasana sepi. Suasana sepi bagi ekstrovert, malah akan membuat semakin tertekan.

- c) Dalam hal percintaan, seorang ekstrovert memiliki keuntungan tersendiri. Seorang introvert bisa bebas berkenalan dengan lawan jenis (*approach*) atau meminta nomor telepon, itu bukan perkara yang sulit bagi mereka. Namun, dalam hal *dating* biasanya seorang ekstrovert memiliki kesulitan. Tipe ekstrovert biasanya lebih sulit untuk membina hubungan *personal* yang lebih dalam dengan seseorang.
- d) Orang ekstrovert terampil dalam melakukan perjalanan ke dunia luar. Dengan leluasa dapat berinteraksi dengan banyak orang. Membuat orang lain terkagum-kagum dan menyukainya. Namun, semua itu dilakukan dengan mengorbankan diri sendiri. Seorang ekstrovert sering terpaksa mengorbankan kepribadiannya sendiri agar dapat diterima oleh orang lain. Pembicaraan orang ekstrovert biasanya bersifat umum.

3. Ambivert

Tipe kepribadian seseorang yang memiliki kedua tipe dasar sehingga sulit untuk memasukkan ke dalam salah satu tipe.

e. Peminatan Karir

Menurut John Holland, dikutip dari laman <http://www.konsultankarir.com/profession/minat-karir> (Diakses 30 April 2017), terdapat enam tipe kepribadian *vocational* yang seringkali disingkat RIASEC. Dengan mengikuti tes ini, seseorang dapat memiliki profil pilihan karir yang unik sesuai dengan minat dan kepribadiannya.

Adapun penjelasan dari ke-6 tipe tersebut adalah sebagai berikut:

1. *Realistic*

Individu dengan minat *realistic* menyukai aktivitas-aktivitas kerja yang bersifat praktis, cepat menangkap masalah dan mencari solusinya. Menikmati bekerja dengan tanaman, hewan, dan material-material lain yang terlihat, seperti kayu, mesin, alat, dll. Menyukai kegiatan luar ruang. Seringkali individu dengan minat *realistic* tidak menyukai pekerjaan yang terutama melibatkan *paper-work* atau pekerjaan yang banyak berhubungan dengan orang lain.

2. *Investigative*

Individu dengan minat *investigative* menyukai aktivitas-aktivitas kerja yang lebih banyak membutuhkan pemikiran mendalam, menyukai bekerja dengan ide dan kekuatan berpikir daripada melakukan aktivitas kerja fisik. Tipe ini menikmati mencari fakta-fakta dan menganalisis masalah secara internal (aktivitas mental) daripada melakukan aktivitas persuasi atau mengarahkan orang lain.

3. *Artistic*

Individu dengan minat *artistic* menyukai aktivitas-aktivitas kerja yang berhubungan dengan sisi artistik dari sesuatu hal atau benda atau obyek, seperti bentuk, desain, dan pola-pola. Menyukai mengekspresikan diri dalam pekerjaan yang dilakukan. Tipe ini lebih suka mengatur dan menyusun pola kerja mereka sendiri tanpa mengikuti seperangkat aturan yang baku.

4. *Social*

Individu dengan minat *social* menyukai aktivitas-aktivitas kerja yang berhubungan dengan individu lainnya. Tipe ini senang membantu dan memajukan orang lain. Selain itu, giat berupaya agar orang tersebut mau mengembangkan diri. Tipe ini lebih suka berkomunikasi dengan orang lain daripada bekerja dengan obyek, mesin, atau data. Tipe ini suka mengajar, memberikan saran, membantu, atau dengan kata lain memberikan pelayanan pada orang lain.

5. *Enterprising*

Individu dengan minat *enterprising* menyukai aktivitas-aktivitas kerja yang bersifat memulai sesuatu atau membangun dari awal (*start-up*), termasuk juga melaksanakan proyek. Tipe ini menyenangi hal-hal yang ‘berbahaya’, terutama dalam bisnis. Disamping itu, tipe ini juga suka meyakinkan dan memimpin orang lain dan senang membuat keputusan. Tipe ini menyukai mengambil resiko untuk mendapatkan keuntungan. Tipe ini lebih menyukai segera mengambil tindakan daripada berpikir mendalam.

6. *Conventional*

Individu dengan minat *conventional* menyukai aktivitas-aktivitas kerja dengan aturan main yang jelas. Tipe ini menyukai prosedur dan standar, dan tidak bermasalah dengan rutinitas. Tipe ini lebih suka bekerja dengan data dan detail daripada bermain dengan ide. Tipe ini juga lebih menyenangi pekerjaan dengan standar yang tinggi dibandingkan harus membuat pertimbangan oleh diri mereka sendiri. Individu dengan tipe ini

menyukai pekerjaan dimana garis wewenang telah ditetapkan dengan jelas.

2.2 Penelitian Terkait

Dalam pelaksanaannya, penulis menggunakan beberapa jurnal penelitian terkait sebagai referensi. Jurnal tersebut mendukung permasalahan yang saat ini diangkat penulis.

Berhasilnya suatu proses belajar akan berbanding lurus dengan tingkat kecerdasan anak. Sayangnya pengetahuan mengenai beragam metode belajar dan memilih metode yang sesuai dengan kriteria anak masih belum banyak dipahami. Kriteria yang mungkin dimiliki adalah Sanguin, Melankolis, Koleris, atau Plegmatis. Sistem Pakar Berbasis Web Mengenai Metode Belajar Anak Berdasarkan Karakteristik Dominan memungkinkan seseorang untuk mengetahui karakter dominan dalam dirinya dengan menggunakan tes psikotes tipe *Personality profile test* dan selanjutnya mendapatkan saran mengenai metode belajar yang sesuai (Cahyaning dan Wahyudin, 2016:150).

Adapun persamaan program yang akan dibuat oleh penulis dengan peneliti Cahyaning dan Wahyudin adalah sebagai berikut:

1. Menggunakan metode pengembangan sistem *waterfall*
2. Menggunakan metode pengembangan pakar dengan *forward chaining*
3. Menggunakan teori pembagian tipe kepribadian yang sama, yaitu teori Hypocrates yang membagi tipe kepribadian menjadi 4, yaitu: Sanguinis, Koleris, Pleghmatis, dan Melankolis.

4. Menggunakan UML dalam perancangan aplikasi dan ERD dalam perancangan basis data.
5. Aplikasi berbasis web

Sedangkan hal yang membedakan aplikasi yang dibuat penulis dengan peneliti Cahyaning dan Wahyudin, peneliti menggunakan karakteristik dominan dalam menentukan metode belajar sedangkan penulis menggunakan model VAK (Visual, Audiometri, Kinestik).

Kepribadian adalah sifat dan tingkah laku khas seseorang yang membedakan seseorang dengan orang lain. Salah satu metode yang digunakan psikolog adalah *Big Five Personality* untuk mendapatkan jenis-jenis kepribadian. Sistem Pakar Kepribadian Diri dengan metode Big Five dapat memudahkan seseorang melakukan tes kepribadian tanpa psikolog, dan memudahkan psikolog dalam melakukan tes kepribadian karena tanpa melakukan perhitungan manual. (Andreyana dkk, 2015:78).

Adapun persamaan program yang akan dibuat oleh penulis dengan peneliti Andreyana dkk adalah sebagai berikut:

1. Menggunakan tema penelitian kepribadian diri yang menjadi salah satu sub-tema yang akan diteliti oleh penulis.
2. Menggunakan metode pengembangan pakar dengan *forward chaining*
3. Aplikasi berbasis web

Sedangkan hal yang membedakan aplikasi yang dibuat penulis dengan peneliti Andreyana dkk adalah sebagai berikut:

1. Teori pembagian tipe kepribadian digunakan oleh peneliti adalah menggunakan metode *Big Five Personality*, sedangkan teori yang digunakan penulis yaitu teori Hypocrates yang membagi tipe kepribadian menjadi 4, yaitu: Sanguinis, Koleris, Pleghmatis, dan Melankolis.
2. Pada tahap desain, peneliti menggunakan permodelan sistem dengan DFD sedangkan penulis menggunakan UML.

Kepribadian sangatlah penting untuk diketahui setiap orang agar setiap individu mampu mengembangkan kelebihan yang dimilikinya. Seseorang yang kesulitan dalam mengembangkan dirinya kemungkinan karena tidak mengetahui sama sekali kelemahan dan kekurangan yang dimilikinya. (Wardiana dan Tobing, 2011:99)

Adapun persamaan program yang akan dibuat oleh penulis dengan peneliti Wardiana dan Tobing adalah sebagai berikut:

1. Menggunakan tema penelitian kepribadian diri yang menjadi salah satu sub-tema yang akan diteliti oleh penulis.
2. Menggunakan UML dalam perancangan aplikasi.
3. Aplikasi berbasis web

Sedangkan hal yang membedakan aplikasi yang dibuat penulis dengan peneliti Wardiana dan Tobing, yaitu peneliti menggunakan 10 (sepuluh) sikap hidup sehari-hari dalam pembahasan kepribadian yaitu kepercayaan pada diri sendiri, optimisme seseorang, tingkat kehati-hatian, ketergantungan, tingkat mementingkan diri sendiri, menilai watak orang lain, daya tahan menghadapi cobaan, toleransi, ambisi, dan empati. Sedangkan teori yang digunakan penulis yaitu teori Hypocrates

yang membagi tipe kepribadian menjadi 4 (empat), yaitu: Sanguinis, Koleris, Pleghmatis, dan Melankolis.

Secara keseluruhan persamaan dari jurnal diatas dengan aplikasi yang akan dibangun penulis adalah tema yang diangkat yaitu seputar kepribadian diri dimana aplikasi tersebut dibuat dengan berbasis web. Adapun perbedaannya, masing-masing peneliti menyajikan satu tes pada penelitiannya sedangkan penulis akan menyajikan beberapa jenis tes dalam aplikasi yang akan dibuat.