BAB II

TINJAUAN PUSTAKA

2.1. Teori Dasar

Supaya penelitian ini dapat berjalan dengan baik, maka diperlukan landasan bagi jalannya penelitian berupa teori yang telah ada. Dalam penelitian ini, akan menjelaskan secara singkat tentang kecerdasan buatan dimana kecerdasan Buatan itu sendiri terdiri dari beberapa cabang yaitu Jaringan Saraf Tiruan, Sistem Pakar dan Logika *Fuzzy*.

2.1.1. Kecerdasan Buatan

Kecerdasan buatan berasal dari bahasa Inggris "*Artificial Intelligence"* disingkat AI, yaitu *Intelligence* adalah kata sifat yang berarti cerdas, sedangkan *Artificial* artinya buatan. Kecerdasan buatan yang dimaksud adalah merujuk pada mesin yang mampu berpikir, mampu mengambil keputusan seperti yang dilakukan manusia. Alan Turing ahli matimatika berkebangsaan inggris yang dijuluki bapak komputer modern dan pembongkar sandi nazi dalam era Perang Dunia ke II 1950, menetapakn defenisi *Artificial Intelegence* "Jika komputer tidak dapat dibedakan dengan manusia saat berbincang melalui terminal komputer, maka bisa dikatakan komputer itu cerdas, mempunyai kecerdasan (Sutojo, 2011). Kecerdasan buatan memungkinkan komputer untuk berfikir dan menirukan proses belajar manusia sehingga informasi baru dapat diserap sebagai pengetahuan dan

proses pembelajaran serta dapat digunakan sebagai acuan di masa yang akan datang.

2.1.2 *Fuzzy logic*

2.1.2.1. Pengertian Logika Fuzzy

Konsep tentang *Fuzzy logic* pertama kali dperkenalkan oleh Prof. Astor Zadeh pada 1962. Logika fuzzy adalah metodologi sistem control pemecahan masalah, yang cocok untuk diimplementasikan pada sistem, mulai dari sistem yang sederhana, sistem kacil, embedded sistem, jaringan PC, multi-chanel atau workstation berbasis akuisisi data, dan sistem control. (Sutojo, 2011, p. 211)

2.1.2.2. Operasi Himpunan Fuzzy

Operasi himpunan *Fuzzy* diperlukan untuk proses inferensi atau penalaran. Dalam hal ini yang dioperasikan adalah derajat keanggotaannya. Berikut beberapa oeprasi dasar yang paling sering digunakan untuk mengombinasi dan memodifikasi himpunan *fuzzy*.

1. Operasi Gabungan *(Union)* sering disebut operator OR dari himpunan fuzzy A dan B, dan dalam operasi gabungan disebut sebagai *Max*.
2. Operasi Irisan *(Intersection)* yang sering disebut dengan operator AND dari himpunan A dan B dan dalam sistem logika *fuzzy*, operasi irisan disebut sebagai *Min.*
3. Operator Komplemen adalah komponen dari himpunan fuzzy A yang sering disebut NOT adalah dengan fungsi keanggotaan untuk setiap x elemen X.(Sutojo, 2011, pp. 227–228)

2.1.2.3.Metode Logica Fuzzy

Metode dari *Logica Fuzzy* sendiri terdiri dari metode diantaranya adalah:

1. Metode Tsukamoto: Secara umum bentuk model *fuzzy* Tsukamoto adalah If(X IS A) and (Y IS B) Then (Z IS C). Dimana A, B dan C adalah himpunan *Fuzzy*. Dalam inferensinya, metode Tsukamoto menggunakan tahapan berikut:
   a. Fuzzyfikasi
   b. Pembentukan basis pengetahuan Fuzzy (Rule dalam bentuk IF..THEN)
   c. Mesin Inferensi
   d. Defuzzyfikasi.
2. Metode Mamdani adalah metode yang paling sering digunakan dalam aplikais karena strukturnya yang sederhana yaitu menggunakan operasi MIN-MAX atau MAX-PRODUCT.
3. Metode Sugeno yaitu output sistem berupa konstanta atau persamaan linier. Metode ini diperkenalkan oleh Takagi-Sugeno Kang pada 1985.(Sutojo, 2011, pp. 233–237)

2.1.2 Pengertian Jaringan Syaraf Tiruan

Jaringan syaraf tiruan adalah paradigma pengelola informasi yang terinspirasi oleh sistem saraf secara biologis, seperti proses informasi pada otak manusia. Elemen kunci dari paradigma ini adalah struktur dari sistem pengolahan informasi yang terdiri yang terdiri dari sejumlah besar elemen pemrosesan yang saling berhubungan (neuron), bekerja serentak untuk menyesaikan masalah

tertentu (Sutojo, 2011, p. 283). Kelebihan-kelebihan yang diberikan oleh JST adalah:

1. Belajar Adaptive, kemampuan untuk mempelajari bagaimana melakukan pekerjaan berdasarkan data yang diberikan untuk pelatihan atau pengalaman awal.
2. *Self-organization*, sebuah JST dapat membuat organisasi sendiri atau representasi dari informasi yang diterimanya selama waktu belajar.
3. *Real-time operation*, perhitungan JST dapat dilakukan secara paralel sehingga perangkat keras yang dirancang dan diproduksi secara khusus dapat mengambil keuntungan dari kemampuan ini.(Sutojo, 2011, p. 284).

Selain mempunyai kelebihan tersebut, JST juga mempunyai kelemahan-kelemahan yaitu sebagai berikut:

1. Tidak efektif untuk melakukan operasi-operasi numeric dengan presisi tinggi
2. Tidak efisien untuk operasi algoritma aritmatik, operasi logika, dan simbolis,
3. Untuk beroperasi JST butuh pelatihan sehingga bila jumlah datanya besar, waktu yang digunakan untuk proses pelatihan sangat lama.(Sutojo, 2011, p. 284).

2.1.3 Sistem Pakar *(Expert Sytem)*

2.1.4.1 Defenisi Sistem Pakar (Expert System)

Sistem pakar adalah salah satu teknik kecerdasan buatan yang meniru penalaran manusia. Pemecahan masalah yang kompleks biasanya hanya dilakukan oleh seorang pakar. Sistem pakar mencoba mencari penyelesaian yang memuaskan, yaitu dengan penyelesaian yang bagus agar pekerjaan dapat berjalan meskipun itu bukan penyelesaian yang optimal (Hartati Sri dan Iswanti Sari, 2008. 2) . Defenisi sistem pakar bisa di lihat pada tabel 2.1.

Tabel 2. 1 Defenisi Sistem Pakar

Sumber	Definisi
Martin dan Oxman (1998)	Sistem berbasis komputer yang menggunakan pengetahuan, fakta, dan teknik penalaran dalam pemecahan masalah, dimana biasanya hanya dapat diselesaikan seorang pakar.
Iqnizio (1991)	Sistem pakar merupakan bidang yang didirikan oleh sistem berbasis pengetahuan (*Knowledge base system*), yang memungkinkan komputer dapat berpikir dan mengambil kesimpulan dari sejumlah kaidah.
Turban dan Aronson (2001)	Sistem yang menggunakan pengetahuan manusia yang kemudian dimasukkan kedalam komputer dan untuk dapat memecahkan masalahnya akan dikerjakan oleh seorang pakar

Sumber : (Hartati Sri dan Iswanti Sari, 2008, p. 3)

2.1.4.2 Kategori Permasalahan Sistem Pakar

Biasanya aplikasi sistem pakar menyentuh beberapa area permasalahan sebagai berikut menurut (Sutojo, 2011, p. 162).

1. Interpretasi, menghasilkan deskripsi situasi berdasarkan data-data masukan.

2. Prediksi, memperkirakan akibat yang mungkin terjadi dari situasi yang ada.
3. Diagnosis, menyimpulkan suatu keadaan berdasarkan gejala yang diberikan (*symptoms*).
4. Desain, melakukan perancangan berdasarkan kendala yang diberikan.
5. *Planning*, merencanakan tindakan yang akan dilakukan.
6. *Monitoring*, membandingkan hasil pengamatan dengan proses perencanaan.
7. *Debugging*, menentukan penyelesaian dari suatu kesalahan sistem.
8. Reparasi, melaksanakan rencana perbaikan.
9. *Instruction*, melakukan instruksi untuk diagnosis, debugging, dan perbaikan kinerja.
10. Kontrol, melakukan kontrol terhadap hasil interpretasi, diagnosis, debugging, monitoring, dan perbaikan tingkah laku sistem.

2.1.4.3 Manfaat dan Kekurangan Sistem Pakar

Menurut (Sutojo, 2011, p. 160) sistem pakar menjadi sangat popular karena sangat banyaknya kemampuan dan manfaat yang diberikan, diantaranya:

1. Meningkatkan produktivitas.
2. Membuat seorang yang awam bekerja seperti layaknya seorang pakar.
3. Meningkatkan kualitas, dengan mengurangi kesalahan.
4. Mampu menangkap pengetahuan dan kepakaran seseorang.
5. Dapat beroperasi di lingkungan yang berbahaya.
6. Memudahkan proses pengetahuan seorang pakar.
7. Andal, sistem pakar tidak pernah menjadi bosan dan kelelahan atau sakit.

8. Meningkatkan kapabilitas sistem komputer..
9. Mampu bekerja dengan informasi yang tidak lengkap atau tidak pasti.
10. Bisa digunakan sebagai media pelengkap dan pelatihan.
11. Meningkatkan kemampuan untuk menyelesaikan masalah.

Selain manfaat, ada juga kekurangan yang ada pada sistem pakar, diantaranya:

1. Biaya yang sangat mahal untuk membuat dan memeliharanya.
2. Sulit dikembangkan karena keterbatasan keahlian dan ketersediaan pakar.
3. Sistem pakar tidak 100% bernilai benar.

2.1.4.4 Komponen Sistem Pakar

Menurut Giarratano dan Riley (2005) dalam (Hartati & Iswanti, 2008, p. 3) sistem pakar sebagai suatu program yang berfungsi untuk menirukan pakar manusia dan harus bisa melakukan hal-hal yang dapat dikerjakan oleh seorang pakar. Untuk membangun sistem yang seperti itu maka komponen yang harus dimiliki adalah sebagai berikut:

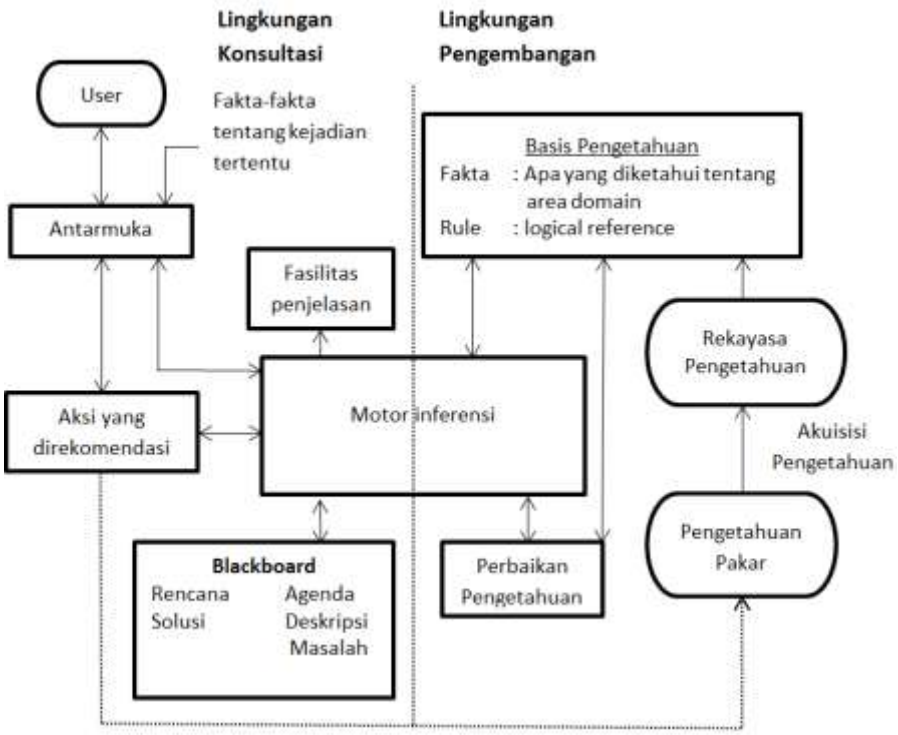
1. Antar muka pengguna (*user interface*),
2. Basis pengetahuan (*knowledge base*),
3. Mekanisme inferensi (*inference machine*),
4. Memori kerja (*working memory*).

Sedangkan untuk menjadikan sistem pakar menjadi lebih menyerupai seorang pakar yang berinterkasi dengan pemakai, maka dilengkapi dengan fasilitas berikut:

1. Fasilitas penjelasan (*explanation facility*),
2. Fasilitas akuisisi pengetahuan (*knowledge acquisition facility*).

2.1.4.5 Struktur Sistem Pakar

Menurut (Sutojo, 2011, p. 166) ada dua bagian penting dari sistem pakar, yaitu lingkungan pengembangan *(development environment)* dan lingkungan konsultasi *(consultation environment)*. Lingkungan pengembangan digunakan oleh pembuat sistem pakar untuk membangun komponen-komponennya dan memperkenalkan pengetahuan ke dalam *knowledgebase* (basis pengetahuan). Lingkungan konsultasi digunakan oleh pengguna untuk berkonsultasi sehingga pengguna mendapatkan pengetahuan dan nasihat dari sistem pakar layaknya berkonsultasi dengan seorang pakar. Gambar 2.1 menunjukkan komponen-komponen yang penting dalam sebuah sistem pakar.



Gambar 2. 1 Komponen-komponen sistem pakar

Sumber: (Sutojo, 2011, p. 166)

Keterangan:

1. Akuisisi Pengetahuan

   Subsistem ini digunakan untuk memasukkan pengetahuan dari seorang pakar dengan cara merekayasa pengetahuan agar bisa di proses oleh *computer* dan menaruhnya ke dalam basis pengetahuan dengan format tertentu (dalam bentuk representasi pengetahuan). Sumber-sumber pengetahuan bisa diperoleh dari pakar, buku.

2. Basis Pengetahuan *(Knowledge Base).*

Basis pengetahuan mengandung pengetahuan yang diperlukan untuk memahami, memformulasikan, dan menyelesaikan masalah. Basis pengetahuan terdiri dari dua elemen dasar, yaitu:

a. Fakta, misalnya situasi, kondisi atau permasalahan yang ada.
b. *Rule* (Aturan), untuk mengarahkan penggunaan pengetahuan dalam memecahkan masalah.

3. Mesin Inferensi *(Inference Engine)*

Mesin inferensi adalah sebuah program yang berfungsi untuk memandu proses penalaran terhadap suatu kondisi berdasarkan pada basis pengetahuan yang ada, memanipulasi dan mengarahkan kaidah, model, dan fakta yang disimpan dalam basis pengetahuan untuk mencapai solusi atau kesimpulan.

4. Daerah Kerja *(Blackboard)*

Untuk merekam hasil sementara yang akan dijadikan sebagai keputusan dan untuk menjelaskan sebuah masalah yang sedang terjadi, sistem pakar membutuhkan *Blackboard*, yaitu area pada memori yang berfungsi sebagai basis data. Tiga tipe keputusan yang dapat direkam pada *blackboard*, yaitu:

a. Rencana: bagaimana menghadapi masalah.
b. Agenda: aksi-aksi potensial yang sedang menunggu untuk dieksekusi.
c. Solusi: calon aksi yang akan dibangkitkan.

5. Antarmuka Pengguna *(User Interface)*

Digunakan sebagai media komunikasi antara pengguna dan sistem pakar. Komunikasi ini paling bagus bila disajikan dalam bahasa alami *(natural*

language) dan dilengkapi dengan grafik, menu, dan formulir elektronik. Pada bagian ini akan terjadi dialog antara sistem pakar dan pengguna.

6. Subsistem Penjelasan *(Explanation Subsystem / Justifier)*

   Berfungsi memberi penjelasan kepada pengguna, bagaimana suatu kesimpulan dapat diambil dan kemampuan ini penting bagi pengguna untuk mengetahui proses pemindahan keahlian pakar maupun dalam pemecahan masalah.

7. Sistem Perbaikan Pengetahuan *(knowledge refining system)*

   Berfungsi memperbaiki pengetahuan *(knowledge refining system)* dari seorang pakar diperlukan untuk menganalisis pengetahuan,dan memperbaiki pengetahuan sehingga dapat dipakai pada masa mendatang.

8. Pengguna *(User)*

   Pada umumnya pengguna sistem pakar bukanlah seorang pakar *(non-expert)* yang membutuhkan solusi, saran, atau pelatihan *(training)* dari berbagai permasalahan yang ada.

2.1.4.6 Representasi Pengetahuan

Pemrosesan yang dilakukan oleh sistem pakar merupakan pemrosesan pengetahuan, bukan pemrosesan data seperti yang dikerjakan dengan pemrograman secara konvensional yang kebanyakan dilakukan oleh sistem informasi. Pengetahuan *(knowledge)* adalah pemahaman secara praktis maupun teoritis terhadap suatu obyek atau domain tertentu (Hartati Sri dan Iswanti Sari, 2008, p. 18). Berikut ini adalah contoh struktur kaidah *IF-THEN* yang

menghubungkan objek (Adedeji, 1992 *dalam* (Hartati Sri dan Iswanti Sari, 2008, p. 25):

1. *IF* premis *THEN* konklusi
2. *IF* masukan *THEN* keluaran
3. *IF* kondisi *THEN* tindakan
4. *IF* antesenden *THEN* konsekuen
5. *IF* data *THEN* hasil
6. *IF* tindakan *THEN* tujuan
7. *IF* aksi *THEN* reaksi
8. *IF* gejala *THEN* diagnosa

Premis mengacu pada fakta yang harus benar sebelum konklusi tertentu dapat diperoleh. Masukan mengacu pada data yang harus tersedia sebelum keluaran dapat diperoleh.Kondisi mengacu pada keadaan yang harus berlaku sebelum tindakan dapat diambil. Data mengacu pada informasi yang harus tersedia sehingga sebuah hasil dapat diperoleh. Tindakan mengacu pada kegiatan yang harus dilakukan sebelum hasil dapat diharapkan. Aksi mengacu pada kegiatan yang menyebabkan munculnya efek dari tindakan tersebut. Gejala mengacu pada keadaan yang menyebabkan adanya kerusakan atau keadaan tertentu yang mendorong adanya pemeriksaan (diagnosa) (Hartati Sri dan Iswanti Sari, 2008, p. 25).

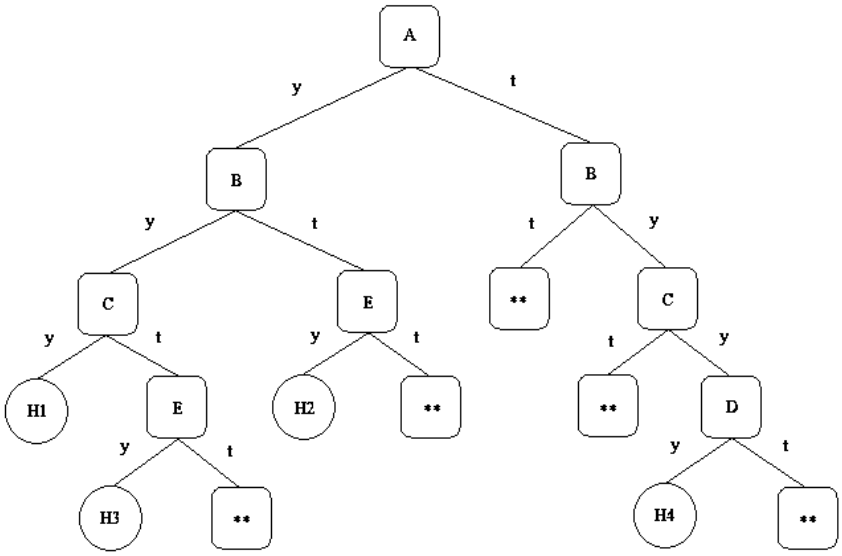
Sebelum sampai pada bentuk kaidah produksi, pengetahuan yang berhasil didapatkan dari domain tertentu disajikan dalam bentuk tabel keputusan kemudian

dibuat pohon keputusannya.Berikut ini adalah contoh penyajian dalam bentuk tabel keputusan dan pohon keputusan (Hartati Sri dan Iswanti Sari, 2008, p. 26)

Tabel 2. 2 Tabel Keputusan

Hipotesa / ***Evidence***	**Hipotesa 1**	**Hipotesa 2**	**Hipotesa 3**	**Hipotesa 4**
Evidence A	Ya	Ya	Ya	tidak
Evidence B	Ya	Tidak	Ya	ya
Evidence C	Ya	Tidak	Tidak	ya
Evidence D	Tidak	Tidak	Tidak	ya
Evidence E	Tidak	Ya	Ya	tidak

Sumber: (Hartati dan Iswanti 2008: 32)



Gambar 2. 2 Pohon Keputusan
Sumber: (Hartati Sri dan Iswanti Sari, 2008, p. 33)

Keterangan:

A = *evidence* A, H1 = hipotesa 1, y = ya

B = *evidence* B, H2 = hipotesa 2, t = tidak

C = *evidence* C, H3 = hipotesa 3, ** = tidak menghasilkan hipotesa tertentu

D = *evidence* D, H4 = hipotesa 4

Dilihat dari gambar 2.3, masing-masing *node* yang mewakili *evidence* tertentu untuk kondisi "y" dan "t" sudah tidak mengarah pada *evidence* yang sama. Hal ini berarti jawaban pengguna yang berbeda akan mengarah pada pertanyaan yang berbeda pula.

Kaidah yang dapat dihasilkan berdasarkan pohon keputusan pada gambar 2.3 adalah sebagai berikut:

1. Kaidah 1: *IF* A *AND* B *AND* C *THEN* H1
2. Kaidah 2: *IF* A *AND* B *AND* E *THEN* H3
3. Kaidah 3: *IF* A *AND* E *THEN* H2
4. Kaidah 4: *IF* D *AND* B *AND* C *THEN* H4

Model representasi pengetahuan kaidah produksi banyak digunakan pada aplikasi sistem pakar karena model representasi ini mudah dipahami dan bersifat deklaratif sesuai dengan jalan pikiran manusia dalam menyelesaikan suatu masalah, dan mudah diinterpretasikan.

2.1.4.7 Mesin Inferensi (Inferensi Engine)

Menurut (Hartati Sri dan Iswanti Sari, 2008, p. 5) mesin inferensi merupakan otak dari sistem pakar, bisa dikatakan sebagai mesin pemikir *(Thinking Machine).* Pada prinsipnya mesin inferensi inilah yang akan mencari solusi dari suatu masalah. Menurut Kusrini (2006: 35) inferensi merupakan proses

untuk menghasilkan informasi dari fakta yang diketahui atau diasumsikan. Inferensi adalah konklusi logis *(logical conclusion)* atau implikasi berdasarkan informasi yang tersedia. Dalam sistem pakar, proses inferensi dilakukan dengan suatu modul yang disebut Mesin Inferensi *(Inference Engine).* Ada dua metode inferensi yang penting dalam sistem pakar, yaitu *forward chaining* (runut maju) dan *backward chaining* (runut mundur).

2.1.4.8 Forward Chaining (Runut Maju)

Menurut (Sutojo, 2011, p. 171) *Forward Chaining* adalah teknik pencarian yang dimulai dengan fakta yang diketahui, kemudian mencocokkan fakta-fakta tersebut dengan bagian *IF* dari *rules IF-THEN*.Bila ada fakta yang cocok dengan bagian *IF*, maka *rule* tersebut dieksekusi. Bila sebuah *rule* dieksekusi, maka sebuah fakta baru (bagian *THEN*) ditambahkan ke dalam *database*.Setiap kali pencocokan, dimulai dari *rule* teratas. Setiap rule hanya boleh dieksekusi sekali saja. Proses pencocokan berhenti bila tidak ada lagi *rule* yang bisa dieksekusi.

Konsep ini dapat juga disebut sebagai pencarian yang dimotori data (*data driven search*). Runut maju melakukan proses perunutan (penalaran) dimulai dari premis-premis atau informasi masukan (*IF*) terlebih dahulu kemudian menuju konklusi atau *derived information* (*THEN*). Konsep ini dapat dimodelkan sebagai berikut:

IF (informasi masukan)

THEN (konklusi)

Informasi masukan dapat berupa suatu pengamatan sedangkan konklusi dapat berupa diagnosa sehingga dapat dikatakan jalannya penalaran runut maju dimulai dari pengamatan menuju diagnosa. Pada metode ini, sistem tidak melakukan praduga apapun terhadap konklusi, namun sistem akan menerima semua gejala yang diberikan pengguna lalu sistem akan memeriksa gejala-gejala tersebut dan selanjutnya mencocokkan dengan konklusi yang sesuai (Hartati Sri dan Iswanti Sari, 2008, p. 45).

2.1.4.9 Validasi Sistem

Menurut (A.S & M.Shalahuddin, 2011, pp. 211–214) validasi mengacu pada sekumpulan aktifitas yang berbeda yang menjamin bahwa perangkat lunak yang dibangun dapat digunakan dan telah sesuai dengan yang diharapkan. Adapun pendekatan dalam melakukan pengujian untuk validasi sistem yaitu:

1. *Black-Box Testing* (Pengujian Kotak Hitam)

   Black-box testing adalah pengujian sistem atau perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program yang bertujuan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari sistem atau perangkat lunak telah sesuai dengan spesifikasi yang dibutuhkan.Pengujian pada *black-box testing* dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan menggunakan sistem apakah telah sesuai dengan spesifikasi yang dibutuhkan.

2. *White-Box Testing* (Pengujian Kotak Putih)

White-box testing adalah pengujian sistem atau perangkat lunak dari segi desain dan kode program apakah mampu menghasilkan fungsi-fungsi, masukan, dan keluaran yang sesuai dengan spesifikasi yang dibutuhkan. Pengujian pada *white-box testing* dilakukan dengan memeriksa logika dari kode program.

2.1 Variabel

Variabel adalah objek penelitian, atau apa saja yang menjadi fokus di dalam suatu penelitian untuk di pelajari dan di peroleh kesimpulannya.(Sudaryono, 2014, p. 16).

Pada penelitian ini, peneliti hanya berfokus pada bagian mesin dari sepeda motor Suzuki nex F1. Peneliti mengangkat gejala kerusakan yang terjadi pada bagian mesin. Beberapa diantaranya yaitu:

1. Kerusakan Piston

Gambar 2. 3 Piston
(**Sumber:** Data Penelitian, 2018)

Piston adalah komponen mesin yang membentuk ruang bakar bersama-sama dengan silinder blok dan silinder head, piston juga dapat melakukan gerakan naik turun untuk melakukan siklus kerja mesin. Piston berfungsi untuk mempertahankan kerapatan antara piston dengan dinding silinder agar tidak ada kebocoran gas dari ruang bakar ke dalam bak mesin. Berikut adalah gejala-gejala yang terjadi pada piston:

1) Mesin motor sulit hidup
2) Tenaga mesin kurang atau lemah
3) Suara mesin kasar
4) Gas motor tersendat-sendat

Cara memperbaiki kerusakan piston yang rusak adalah harus mengganti piston tersebut. Namun sebelum menggantinya pastikan bahwa piston yang baru sesuai dengan spesifikasi motor Suzuki nex F1.

2. Kerusakan klep

Gambar 2. 4 Klep

(**Sumber:** Data Penelitian, 2018)

Klep atau yang sering juga dikenal sebagai katup adalah salah satu komponen pada mesin motor yang mempunyai sifat dinamis yang biasanya terpasang dikepala silinder. Fungsi katup adalah untuk mengatur masuknya gas baru dan keluarnya gas buang sisa pembakaran pada mesin motor. Gejala yang terjadi akibat kerusakan pada klep motor yaitu:

1) Mesin motor sulit hidup.
2) Motor mengeluarkan asap tebal berwarna hitam.
3) Oli mesin cepat habis.
4) Bahan bakar motor menjadi boros.

Solusinya adalah membersihkan klep, *siting* dan bos klep menggunakan bensin, Bila tidak terjadi perubahan gantilah katup tersebut dengan yang baru dan melakukan pemasangan dengan hati-hati.

3. Kerusakan Karburator

Gambar 2. 5 Karburator
(**Sumber**: Data Penelitian, 2018)

Karburator merupakan komponen untuk mencampur udara dan bahan bakar. Fungsi karburator ada dua yaitu pertama, mengatur RPM untuk mencampur udara dan bahan bakar sesuai dengan perbandingan. Kedua, karburator juga harus mampu menyumplai bensin dengan perbandingan yang ideal pada segala RPM. Gejala-gejala yang terjadi pada karburator yaitu:

1) Mesin motor sulit hidup.
2) Motor mengeluarkan asap tebal berwarna hitam.
3) Knalpot mengeluarkan suara (knalpot nembak)
4) Kecepatan motor sulit untuik bertambah

Solusinya adalah putar setelan sekrup udara searah jarum jam atau arah menutup, yaitu memaksudkan memperbanyak campuran udara dengan bahan bakar. Kemudian bersihan karburator dari kotoran yang menempel.

4. Kerusakan Busi

Gambar 2. 6 Busi

(**Sumber**: Data Penelitian, 2018)

Busi memiliki peranan penting dalam sistem pengapian pada kendaraan sepeda motor, busi berfungsi untuk mengubah tegangan listrik dengan yang disalurkan koil, menjadi menjadi percikan api yang akan membakar campuran bahan bakar dan udara diruang mesin. Kerusakan yang terjadi akibat busi rusak yaitu:

1) Mesin motor sulit hidup.
2) Tenaga mesin kurang atau lemah.
3) Suara mesin kasar.
4) Mesin mati mendadak ketika dijalan.

Solusinya adalah membersihkan kerak-kerak hitam yang menempel pada bagian kepala busi, jika masih tetap rusak, maka busi perlu diganti yang baru.

5. Kerusakan Injektor

Gambar 2. 7 Injektor

(**Sumber**: Data Penelitian, 2018)

Mempunyai fungsi yaitu untuk metode pencampuran bahan bakar dengan udara pada kendaraan bermotor untuk menghasilkan pembakaran yang sempurna. Kerusakan yang disebabkan oleh injector pada motor yaitu:

1) Mesin sulit distater,
2) Asap knalpot lebih pekat
3) Mesin sering mati mendadak.
4) Mesin mogok dan tidak mampu dihidupkan

Solusi untuk memperbaiki dari kerusakan injector adalah maka perlu mengecek apakah bensin sampai keruang bakar, Cek bagian–bagian yang terhubung dengan *injector*, bila tidak ada perubahan *g*antilah *injector* dengan yang baru.

2.2 Software Pendukung

2.3.1. *UML (Unified Modelling Language)*

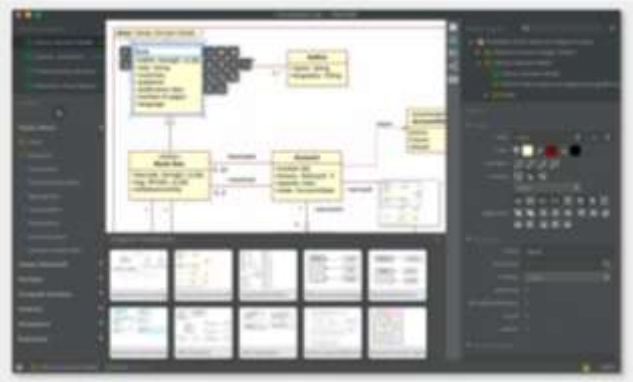
UML *(Unified Modeling Language)* adalah sebuah standarisasi bahasa pemodelan perangkat lunak yang dibangun menggunakan teknik pemprograman berorientasi objek. UML muncul karena adanya kebutuhan pemodelan visual yang menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak.

UML *(Unified Modeling Language)* juga berarti bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metedeologi tertentu,

meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek (A.S & M.Shalahuddin, 2011, p. 135). Salah satu aplikasi untuk memudahkan menggunakan UML adalah StarUML.

Gambar 2. 8 Logo StarUML
(**Sumber:** http://staruml.sourceforge.net/image/staruml-logo.jpg)

StarUML merupakan salah satu *CASE (Computer-Aided Software Engineering) tools* atau perangkat pembantu berbasis komputer untuk rekayasa perangkat lunak yang mendukung alur hidup perangkat lunak *(life cycle support). StarUML* termasuk ke dalam kelompok *upperCASE tools* yang mendukung perencanaan strategis dan pembangunan perangkat lunak (Rosa & M.Shalahuddin, 2013, pp. 122–123).



Gambar 2. 9 Interface StarUML
(**Sumber** : http://staruml.io)

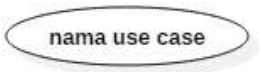
UML mendefenisikan diagram-diagram adalah sebagai berikut:

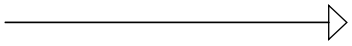
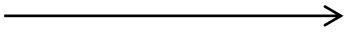
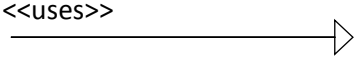
1. *Use case diagram*
2. *Activity diagram*
3. *Sequence diagram*
4. *Class diagram*

2.3.1.1 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana" . *Use case diagram* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya (Yasin, 2012: 198).

Tabel 2. 3 Simbol *Use Case Diagram*

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
Aktor/*actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama actor
asosiasi/*association* ———	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor

Ekstensi/*extend* <<extend>> 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walaupun tanpa *use case* tambahan itu. Arah panah mengarah pada *use case* yang ditambahkan.
generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara 2 buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari fungsi lainnya. Arah panah mengarah pada *use case* yang menjadi generalisasinya (umum)
Menggunakan/*include*/*uses* <<include>>  <<uses>> 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankannya *use case* ini. Arah panah mengarah pada *use case* yang ditambahkan

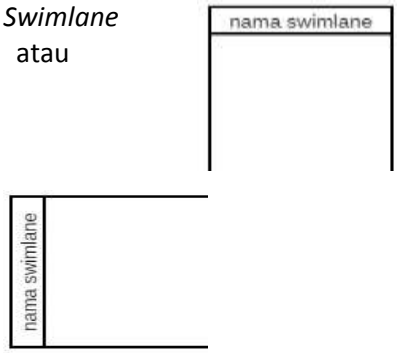
Sumber: (A.S & M.Shalahuddin, 2011, p. 156)

2.3.1.2 Activity Diagram

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktifitas dari sebuah sistem atau proses bisnis atau manu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukam apa yang dilakukan aktor, jadi aktifitas yang dapat dilakukan oleh sistem.

Tabel 2. 4 Simbol *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktifitas memiliki sebuah status awal
Aktifitas 	Aktifitas yang dilakukan sistem, aktifitas biasanya diawali dengan kata kerja

Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu
Penggabungan/*join* 	Asosiasi penggabungan dimana lebih dari satu aktifitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktifitas memiliki sebuah status akhir
Swimlane atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi

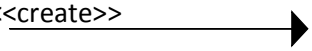
Sumber: (A.S & M.Shalahuddin, 2011, pp. 162–163)

2.3.1.3 Sequence Diagram

Menurut (A.S & M.Shalahuddin, 2011, p. 137) diagram sequen menggambarkan kelakuan objek pada *use case* dengan mendekripsikan waktu hidup objek dan *massage* yang dikirimkan dan diterima antar objek.

Tabel 2. 5 Simbol *Sequence Diagram*

Simbol	Deskripsi
Aktor/*actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama actor
Garis hidup/*lifeline* 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor

Objek nama objek: nama kelas	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya. Aktor tidak memiliki waktu aktif
Pesan tipe *create* <<create>> 	Menyatakan suatu objek membuat objek yang lain. Arah panah mengarah pada objek yang dibuat

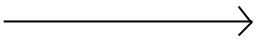
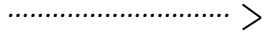
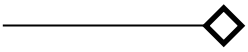
Sumber: (A.S & M.Shalahuddin, 2011, p. 209)

2.3.1.4 Class Diagram

Menurut (A.S & M.Shalahuddin, 2011, p. 122) *Class Diagram* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain beorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi).

Tabel 2. 6 Simbol *Class Diagram*

Simbol	Deskripsi
Kelas Nama_kelas +atribut +operasi()	Kelas pada struktur sistem
Antarmuka / *interface*  Nama_*interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.

Asosiasi / *association* 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga diserta dengan *multiplicity*
Asosiasi berarah / *directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antar kelas dengan makna generalsasi-spesialisasi (umum-khusus)
Kebergantungan / *dependency* 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi / *aggregation* 	Semua-bagian (*whole-part*)

Sumber: (A.S & M.Shalahuddin, 2011, pp. 123–124)

2.3.1 *Android*

Android merupakan *software* yang digunakan pada perangkat *mobile* yang mencakup sistem operasi, *middleware,* dan aplikasi kunci yang dirilis oleh *Google*. Sehingga android mencakup keseluruhan sebuah aplikasi, mulai dari sistem operasi sampai pada pengembangan aplikasi itu sendiri. Pengembangan aplikasi pada platform *Android* ini menggunakan dasar bahasa pemprograman Java. Platform pengembangan aplikasi *Android* ini bersifat *open-source* atau terbuka, sehingga dapat dapat mengembangkan kemampuan untuk membangun

aplikasi yang kaya dan inovatif.(EMS, 2015). Fitur yang terdapat dalam android antara lain adalah:

1. *Android run-time*, yaitu terdiri atas *library java* dan *dalvik virtual machine*.
2. *Open graphics library*, yaitu *application program interface* yang digunakan untuk membuat grafis 2D dan 3D.
3. *Webkit*, yaitu *engine* dari *web browser* yang digunakan untuk menampilkan isi *website* dan menyederhanakan tampilan dari proses *loading*.
4. SQLite, yaitu *engine* dari relasional *database* yang dapat diintegrasikan dengan aplikasi.

Menurut (EMS, 2015) Kelebihan dari penggunaan sistem *android*, yaitu:

1. *Multitasking*, dapat menjalankan aplikasi secara bersamaan.
2. Terdapat notifikasi ketika ada panggilan atau sms, yaitu ketika ada sms atau *email* yang masuk, akan terdapat notifikasi pada ponsel.
3. Dukungan ribuan aplikasi terpercaya melalui situs *Google Play*, yaitu berfungsi mendapatkan berbagai aplikasi yang diperlukan.
4. Penggunaan *widget* pada *home screen*, yaitu memudahkan dan mempercepat pengguna ketika membuka aplikasi.

Gambar 2. 10 Logo Android
(**Sumber**: (EMS, 2015, p. 2)

2.3.4 Java

Java adalah bahasa pemrograman yang dapat dijalankan di komputer maupun telepon genggam. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam *p-code (bytecode)* dan dapat dijalankan pada berbagai *Mesin Virtual Java (JVM).* Java merupakan bahasa pemrograman yang bersifat umum dan secara khusus didesain untuk memanfaatkan dependensi implementasi seminimal mungkin (Bambang Hariyanto, 2014: 4).

Karena fungsionalitasnya yang memungkinkan aplikasi *java* mampu berjalan dibeberapa *platform* sistem operasi yang berbeda. *Java* memiliki beberapa kelebihan yaitu :

1. Bahasa sederhana.
2. Bahasa orientasi objek.
3. Bahasa *statically Typed.*
4. Bahasa dikompilasi.
5. Bahasa yang aman.
6. Bahasa *independen* terhadap *platform.*
7. Bahasa *multithreading.*
8. Bahasa yang didukung *garbage collector.*
9. Bahasa yang tangguh.
10. Bahasa yang mampu diperluas.

Gambar 2. 11 Logo Java
(**Sumber**: Bambang Hariyanto, 2014: 2)

2.3.4 *Eclipse*

Eclipse adalah tool editor yang berdiri sendiri yang dipakai dalam pemprograman android baik berbasis java atau dengan pemaketan HTML (EMS, 2015, p. 35).

Gambar 2. 12 Logo Eclipse
Sumber : (EMS, 2015, p. 37)

2.4 Penelitian Terdahulu

Penelitian pertama yang dilakukan (Anggraheni & Uly Wartaty, 2014) yang berjudul ***"Sistem Pakar Untuk Mendiagnosa Kerusakan Sepeda Motor Non Injeksi Pada Bengkel Gemilang Jaya Motor Kabupaten Pacitan".*** Pada penelitian ini, sistem pakar untuk mendeteksi kerusakan pada sepeda motor *non* injeksi ini merupakan suatu sistem untuk mempermudah pemilik motor mendeteksi kerusaskan pada sepeda motor. Sehingga pemilik dapat mengetahui lebih dini kerusakan pada sepeda motor dan dapat melakukan tindakan awal

sebelum ditindak lanjuti oleh mekanik ataupun dapat menangani kerusakan-kerusakan ringan pada sepeda motor.

Penelitian kedua dilakukan oleh (Maria Shusanti F, 2010) yang berjudul **"Sistem Pakar Untuk Mendeteksi Kerusakan Pada Sepeda Motor 4-tak Dengan Menggunakan Metode Backward Chaining"** : Kendaraan sepeda motor merupakan suatu alat transportasi yang banyak digunakan masyarakat pada umumnya. Pertumbuhan jumlah sepeda motor sangat pesat seiring dengan tingkat ekonomi dan kebutuhan masyarakat terhadap alat transportasi yang murah dan terjangkau sehingga banyak masyarakat yang memilih sepeda motor 4-tak sebagai kendaraan pribadinya untuk digunakan dalam aktivitas sehari-hari. Penelitian ini akan menggambarkan serta menjelaskan sistem pakar yang akan digunakan untuk mendeteksi kerusakan sepeda motor 4-tak dengan menggunakan metode backward chaining dengan jelas. Peneliti akan menggunakan metode *backward chaining* sebagai metode inferensi untuk sistem pakar ini. sistem pakar untuk mendeteksi kerusakan sepeda motor 4-tak dengan menggunakan metode *backward chaining* dapat menggantikan tenaga ahli atau pakar dalam mendeteksi kerusakan sepeda motor 4-tak sehingga kerusakan tersebut dapat diatasi dengan cepat. Selain itu dengan adanya fitur untuk menambah pengetahuan yang hak aksesnya diberikan kepada pengguna pakar maka sistem pakar ini akan dapat terus berkembang sesuai dengan keadaan dan kebutuhan. Kelebihan dari penggunaan metode *backward chaining* adalah metode ini memiliki sifat yang cocok dalam melakukan diagnosa dan dalam pencarian solusi menggunakan metode ini tidak memerlukan waktu yang lama bila pengguna memberikan

informasi yang tepat, tetapi sebaliknya metode ini memiliki kelemahan dalam pencarian solusi, bila pemilihan hipotesa atau kesimpulan tidak tepat maka si pengguna tidak dapat menemukan langsung solusi yang dibutuhkannya sehingga pengguna harus berulang kali melakukan konsultasi.

Penelitian ketiga dilakukan oleh (Sodiq & Shinta, 2016) dengan judul **"Rancang bangun sistem pakar untuk Diagnosa kerusakan pada motor matic dengan Metode *Fordward Chaining".*** Penelitian ini menekankan pada diagnosis dari masalah yang terjadi pada sepeda motor, khususnya teknologi Honda Vario CVT atau lebih dikenal dengan sebutanmotor matic untuk mengkhususkan pada transmisi / CVT. Metode pengumpulandata penelitian mulai menggunakan metode observasi (observasi), dan literatur atau yang diketahui literatur. Desain sebuah sistem pakar untuk mendeteksi kerusakan pada motor dibuat dengan menggunakan metode *forward chaining* dengan Bahasa pemrograman Microsoft Visual Basic adalah bahasa pemrograman yang dibuat Microsoft Access. Sistem pakar membuatnya mudah pengguna motor matic untuk menentukan kerusakan pada motor maticnya secara mendalam dan menentukan segala jenis sparebagian yang harus diganti dengan penerapangejala kerusakan yang tepat. Buka mereka pengetahuan tentang komponen apa saja yang terdapat pada motor matic dan akhirnya konsumen merasa puasdengan aplikasi tersebut.

Penelitian keempat yang di lakukan oleh (Anggri Sartika Wiguna, 2017) **"Sistem pakar Diagnosa Kerusakan Sepeda Motor Matic Injeksi menggunakan metode Forward Chaining berbasis android".** Perkembangan industri sepeda motor *matic* injeksi di indonesia mengalami perkembangan yang

signifikan, sepeda motor matic injeksi yang lebih irit bahan bakar dan ramah lingkungan, dengan tingginya pengguna sepeda motor matic injeksi saat ini timbul permasalahan

bahwa tidak semua pengguna motor matic injeksi memiliki kemampuan melakukan perbaikan terhadap kerusakan sepeda motornya. Dengan kemajuan teknologi smarthphone saat ini, memunculkan suatu ide atau gagasan aplikasi sistem pakar ke dalam aktivitas mutu pelayanan smartphone. Sistem yang akan dibuat adalah "Sistem pakar diagnosa kerusakan sepeda motor matic injeksi menggunakan metode *forward chaining* berbasis mobile" aplikasi ini akan menggunakan metode *forward chaining*. Sistem pakar ini diharapkan dapat membantu pengguna mengetahui kerusakan dan melakukan perbaikan sepeda motornya lebih awal sebelum terjadi kerusakan yang berkelanjutan. Sistem

ini dibangun menggunakan aplikasi *Basic4 android*.

Penelitian kelima dari jurnal Internasional yang di teliti oleh (Martinez & Ramirez, 2014) ***"Using an improved rule match algorithm in an expert system detect broken driving rules for an energy-efficiency and safety relevant driving system"*** *yaitu Vehicles have been so far improved in terms of energy-efficiency and safety mainly by optimising the engine and the power train. However, there are opportunities to increase energy-efficiency and safety by adapting the individual driving behaviour in the given driving situation. In this paper, an improved rule match algorithm is introduced, which is used in the expert system of a human-centred driving system. The goal of the driving system is to optimise*

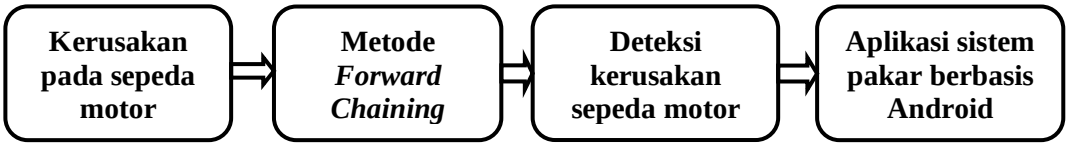
the driving behaviour in terms of energy-efficiency and safety by giving recommendations to the driver. The improved rule match algorithm checks the incoming information against the driving rules to recognise any breakings of a driving rule. The needed information is obtained by monitoring the driver, the current driving situation as well as the car, using in-vehicle sensors and serial-bus systems. On the basis of the detected broken driving rules, the expert system will create individual recommendations in terms of energy-efficiency and safety, which will allow eliminating bad driving habits, while considering the driver needs.

2.5 Kerangka Pemikiran

Menurut Sekaran (2006) dalam (Sudaryono, 2014) mengemukakan bahwa kerangka berpikir adalah model konseptual tentang bagaimana teori berhubungan denganberbagai faktor yang telah diidentifikasi sebagai masalah penting. Secara teoritis kerangka berpikir yang baik akan menjelaskan peraturan antar variabel yang akan diteliti. Kerangka berpikir di dalam penelitian perlu dikemukakan apabila penelitian hanya membahas satu variabel atau lebih secara mandiri, disamping mengemukakan deskripsi teoritis untuk masing-masing varibel, peneliti juga harus menyatakan argumentasi terhadap besaran variabel yang diteliti. (Sudaryono, 2014). Selanjutanya dalam mengemukakan bahwa kerangka berpikir yang baik memuat hal-hal berikut:

1. Variabel yang di teliti harus dijelaskan.

2. Diskusi dalam kerangka berpikir harus dapat menunjukkan dan menjelaskan hubungan atau keterkaitan antar variabel yang di teliti, sreta teori yang mendasarinya.
3. Diskusi juga harus dapat menujukkan dan menjelaskan apakah hubungna antar variabel itu positif atau negatif, berbentuk simetris, kausal, atau interaktif (timbale balik atau umpan balik)
4. Kerangka berpikir selanjutnya dinyatakan dalam bentuk diagram sehingga pihak lain dapat memahami kerangka berpikur yang dikemukakan dalam penelitian.Berikut adalah kerangka berpikir dari penelitian ini. (Sudaryono, 2015:21)



Gambar 2. 13 Kerangka Pemikiran
Sumber: Olahan Peneliti (2018)