BAB II
LANDASAN TEORI

2.1. Teori Dasar

2.1.1 Kecerdasan Buatan (*Artificial Intelligence*)

Keceradasn Buatan atau *Artificial Intelligence* (AI), *Intelligence* yang berarti kecerdasan, *Artificial* berarti buatan. Kecerdasan buatan yang dimaksud di sini mengacu pada mesin yang mampu berpikir, mengukur tindakan yang diambil dan membuat keputusan.

Kecerdasan Buatan adalah kecerdasan yang ditambahkan pada suatu sistem yang bisa diatur dalam konteks ilmiah atau bisa disebut juga intelegensi artifisial, *Artificial Intelligence* (AI) dalam Bahasa inggris. Kecerdasan Buatan juga merupakan kemampuan sistem untuk menafsirkan data eksternal dengan benar, untuk belajar dari data tersebut, dan menggunakan pembelajaran tersebut guna mencapai tujuan dengan melalui adaptasi yang fleksibel.(Kaplan & Haenlein, 2019). Berikut berupa metode-metode Keceradasan Buatan (AI):

2.1.1.1. Sistem Pakar atau *Expert System*

Sistem Pakar atau *Expert System* adalah sistem computer yang mengemulasi kemampuan pengambilan keputusasn seorang ahli manusia, Sistem pakar dirancang untuk memecahkan masalah yang kompleks dengan bernalar melalui kumpulan pengetahuan, yang diwakili terutama seolah-olah memerintah daripada melalui kode prosedural konvensional, sistem pakar yang *modern* dapat

ditingkatkan dengan penambahan basis pengetahuan atau set aturan seperti permainan catur dan sistem diagnosis medis (Nwigbo Stella & Agbo Okechuku Chuks, 2015).

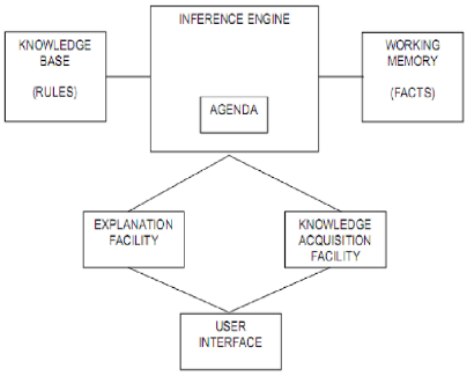
a. **Keuntungan Sistem Pakar**

Secara garis besar, banyak manfaat yang dapat di ambil dengan adanya sistem pakar, antara lain (Kusumadewi, 2003:110):

1. Memungkingkan orang awam untuk melakukan pekerjaan para ahli
2. Menyimpan pengetahuan dan keahlian pakar
3. Meningkatkan output dan produktivitas
4. Meningkatkan kualitas
5. Memiliki realiabilitas
6. Meningkatkan kapabilitas sistem computer
7. Sebagai media pelengkap dalam pelatihan
8. Permanen
9. Memiliki kemampuan untuk mengakses pengetahuan

b. **Struktur Sistem Pakar**

Adapun struktur sistem pakar dapat dilihat pada gambar berikut (Rosnelly, 2012).



Gambar 2.1 Struktur Sistem Pakar

Berikut berupa komponen yang terdapat dalam struktur sistem pakar:

1. *Knowledge Base* (Base Pengetahuan)

Adalah program komputer yang beralasan dan menggunakan basis pengetahuan untuk memecahkan masalah yang kompleks. Istilah ini luas dan mengacu pada berbagai jenis sistem. Satu tema umum yang menyatukan semua sistem berbasis pengetahuan adalah upaya untuk mewakili pengetahuan secara eksplisit dan sistem penalaran yang memungkinkannya untuk memperoleh pengetahuan baru.

2. *Inference Engine* (Mesin Inferensi)

adalah komponen sistem yang menerapkan aturan logis ke basis pengetahuan untuk menyimpulkan informasi baru. Mesin inferensi pertama adalah komponen sistem pakar. Mesin inferensi menerapkan aturan logis ke basis pengetahuan dan menyimpulkan pengetahuan baru. Proses ini akan berulang karena setiap fakta

baru dalam basis pengetahuan dapat memicu aturan tambahan dalam mesin inferensi. Mesin inferensi bekerja terutama di salah satu dari dua mode baik aturan khusus atau fakta, yaitu *forward chaining* dan *backward chaining*.

3. *Working Memory*

Untuk menyimpan data dari *inference engine*.

4. *Explanation Facility*

Modul yang memungkinkan sistem pakar untuk memberikan penjelasan kepada pengguna tentang mengapa ia mengajukan pertanyaan dan bagaimana hal itu mencapai beberapa kesimpulan.

5. *Knowledge Acquisition Facility*

Mencakup proses memasukan informai-informasi seorang pakar/ahli ke dalam database program, untuk memperbaiki ataupun mengembangkan basis pengetahuan.

6. *User Interface*

Antarmuka untuk dilihat dan di pakai oleh pengguna sistem dan mengubahnya menjadi data yang dapat diterima oleh sistem. Antarmuka juga menerima informasi dari sistem dan muncul dalam wujud yang pengguna bisa mengerti.

c. ***Inference Engine* atau Mesin Interferensi**

Mesin inferensi adalah komponen sistem yang menerapkan aturan logis ke basis pengetahuan untuk menyimpulkan informasi baru. Mesin inferensi pertama

adalah komponen sistem pakar. Sistem pakar yang khas terdiri dari basis pengetahuan dan mesin inferensi. Basis pengetahuan menyimpan fakta tentang dunia. Mesin inferensi menerapkan aturan logis ke basis pengetahuan dan menyimpulkan pengetahuan baru. Proses ini akan berulang karena setiap fakta baru dalam basis pengetahuan dapat memicu aturan tambahan dalam mesin inferensi. Mesin inferensi bekerja terutama di salah satu dari dua mode baik aturan khusus atau fakta: forward chaining dan backward chaining. Forward chaining dimulai dengan fakta yang diketahui dan menegaskan fakta baru. Rantai mundur dimulai dengan tujuan, dan bekerja mundur untuk menentukan fakta apa yang harus ditegaskan agar tujuan dapat dicapai(Lenat D.B. & Weyer, 2016).

Ada dua cara inferensi di dalam sistem pakar yaitu:

1. Rumut Maju(*Forward Chaining*)

Forward chaining adalah pencocokan data atau pernyataan mulai dari sisi kiri (*IF* dulu). Penalaran dimulai dari fakta terlebih dahulu untuk menguji kebenaran hipotesis (Octavina & Fadlil, 2014).

2. Rumut Balik(*Backward Chaining*)

Backward chaining adalah adalah pencocokan data atau pernyataan mulai dari sisi kiri (*THEN* dulu). Penalaran dimulai dari hipotesa terlebih dahulu. Dan untuk menguji fakta hipotesis harus ditemukan di basis pengetahuan (Octavina & Fadlil, 2014).

d. **Karakteristik Sistem Pakar.**

Sistem pakar memiliki beberapa karakteristik umum, berupa kinerja yang baik, waktu respons yang cepat, sangat andal, mudah dipahami, dan fleksibel.

2.1.1.2. Jaringan Syaraf Tiruan

adalah jaringan dari sekelompok unit pemroses kecil yang dimodelkan berdasarkan sistem saraf manusia. JST merupakan sistem adaptif yang dapat mengubah strukturnya untuk memecahkan masalah berdasarkan informasi eksternal maupun internal yang mengalir melalui jaringan tersebut. Oleh karena sifatnya yang adaptif, JST juga sering disebut dengan jaringan adaptif (Irwansyah & Faisal, 2015).

Secara sederhana, JST adalah sebuah alat pemodelan data statistik non-linier. JST dapat digunakan untuk memodelkan hubungan yang kompleks antara input dan output untuk menemukan pola-pola pada data. Menurut suatu teorema yang disebut "teorema penaksiran universal", JST dengan minimal sebuah lapis tersembunyi dengan fungsi aktivasi non-linear dapat memodelkan seluruh fungsi terukur Boreal apapun dari suatu dimensi ke dimensi lainnya (Irwansyah & Faisal, 2015). Proses belajarnya JST berbeda dengan metode yang lain, dimana *Neural Network* memiliki struktur yang distingtif, berikut adalah kelompok metode pelatihannya:

a. *Feed Forward Neural Network (FFNN)*

Feed Forward Neural adalah jenis spesifik dari jaringan saraf tiruan awal yang dikenal karena kesederhanaan desainnya. Jaringan neural feedforward memiliki lapisan input, lapisan tersembunyi dan lapisan output. Informasi selalu bergerak dalam satu arah, dari lapisan input ke lapisan output dan tidak pernah bergerak ke belakang (Irwansyah & Faisal, 2015).

b. *Feed Back Neural Network (FBNN)*

Feed Back Neural Network adalah kelas jaringan saraf tiruan di mana koneksi antara node membentuk grafik diarahkan sepanjang urutan. Ini memungkinkannya untuk menunjukkan perilaku dinamis temporal untuk urutan waktu. Tidak seperti jaringan neural feedforward, RNNs dapat menggunakan status internal (memori) untuk memproses urutan input. Ini membuatnya dapat diterapkan untuk tugas-tugas seperti pengakuan tulisan tangan yang tidak teregmentasi, terhubung atau pengenalan ucapan (Irwansyah & Faisal, 2015).

2.1.1.3. Logika *Fuzzy*

Logika *Fuzzy* adalah cabang dari *soft computing*. Logika *Fuzzy* adalah suatu teori kompilasi logika yang terus dikembang supaya dapat megantasi konsep nilai yang ada diantara benar dan salah (Irwansyah & Faisal, 2015). Logika *Fuzzy* terdapat 3 metode, yaitu:

a. Metode Mamdani

Metode Mamdani sering juga dikenal dengan nama metode MIN-MAX. berikut beberapa tahap untuk mendapatkan output (Pusadan, 2014) :

1. Penyusunan Himpunan *fuzzy*
2. Aplikasi Fungsi Implikasi
3. Komposisi Aturan
4. Defuzzifikasi

b. Metode Sugeno

Berupa penalaran Bersama cara *output* sistem, bukan himpunan *fuzzy*, tetapi konstanta atau persamaan lienar (Rofiq, 2013).

c. Metode Tsukamoto

Semua *rule* menggunakan himpunan-himpunan *fuzzy*, dengan fungsi keangotaan yang menonton, Untuk mendapatkan hasil yang benar, itu dicari dengan memodifikasi input pengunaan menjadi angka di *domain* yang ditetapkan oleh *fuzzy* (Sholihin, Fuad, & Khamiliyah, 2013).

2.1.2 Tabel Keputusan

Tabel yang digunakan sebagai alat bantu untuk menyelesaikan logika dalam program. Algoritma yang berisi keputusan bertingkat yang banyak sekali sangat sulit untuk digambarkan langsung dengan structured English atau pseudocode dan dapat dibuat terlebih dahulu dengan menggunakan tabel keputusan. Dengan

demikian tabel keputusan efektif digunakan bilamana kondisi yang akan diseleksi didalam program jumlahnya cukup banyak dan rumit. (Herjanto & Herfan, 2008).

Tabel 2.1 Tabel Keputusan

	Hipotesa 1	**Hipotesa 2**	**Hipotesa 3**	**Hipotesa 4**
Indikator A	Y	Y	N	Y
Indikator B	Y	N	Y	N
Indikator C	Y	N	Y	N
Indikator D	N	Y	N	Y

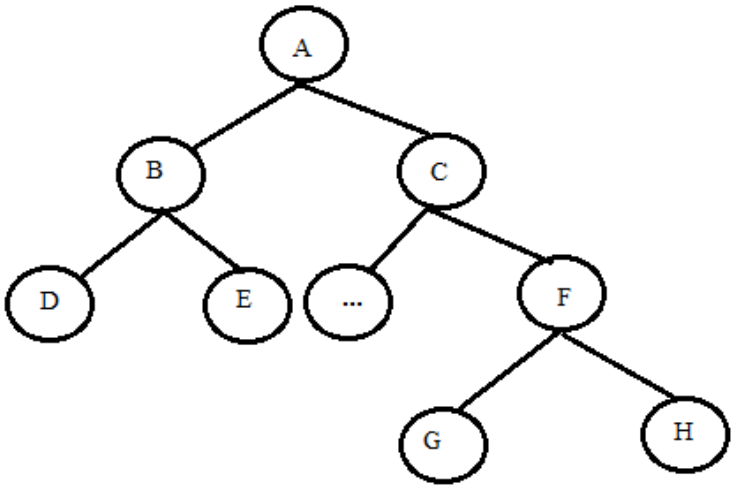
2.1.3 Pohon Keputusan

Pohon keputusan adalah salah satu metode klasifikasi yang paling populer karena mudah untuk di interpretasi oleh manusia. Pohon keputusan adalah model prediksi menggunakan struktur pohon atau struktur berhirarki. Pohon keputusan adalah salah satu metode klasifikasi yang paling populer karena mudah untuk di interpretasi oleh manusia. Pohon keputusan adalah model prediksi menggunakan struktur pohon atau struktur berhirarki (Kusrini & Luthfi, 2009).

Manfaat utama dari menggunakan pohon keputusan adalah kemampuan mereka untuk memecah proses pengambilan keputusan yang kompleks menjadi lebih sederhana sehingga pengambil keputusan akan lebih baik menafsirkan solusi untuk masalah tersebut. Decision Tree juga berguna untuk mengeksplorasi data,

menemukan hubungan tersembunyi antara sejumlah variabel input prospektif dengan variabel target (Kusrini & Luthfi, 2009).

Pohon keputusan harus dirancang hati-hati dengan cara manual atau secara otomatis dengan memakai satu atau lebih algoritme pohon keputusan untuk membuat kumpulan data tanpa diklasifikasikan.



Gambar 2.2 Pohon Keputusan

2.2. Variabel Penelitian

Variabel penelitian merupakan bagian krusial dari penelitian. Kemampuan peneliti dalam mengerti variable penelitian sangat tergantung pada pahamnya konsep dalam penelitian, terpenting adalah variable penelitian. Pengalaman peneliti dalam melakukan penelitian dan menyiapkan proposal penelitian juga dapat membantu pengetahuan tentang kemampuan untuk menentukan variable penelitian. Variable merupakan rancangan yang dioperasionalkan. (Swarjana, 2015).

Variabel yang digunakan peneliti adalah sebagai berikut:

2.2.1 Mesin Manufaktur

Indikator kerusakan yang terdapat pada mesin manufaktur adalah sebagai berikut:

1. Pompa air pendingin

   Adalah tipe pompa yang diapaki untuk mensirkulasi pendingin, biasanya dalam bentuk air (Singh, 2006:148).

2. Minyak Pelumas

   Adalah sebuah zat yang dipakai untuk mengurangi gesekan antara dua permukaan yang saling bersentuhan. (Singh, 2006:290).

3. Sparepart

   Adalah unsur yang bisa dipertukarkan dan disimpan dalam inventaris lalu dipakai untuk perbaikan atau penggantian unit gagal (Singh, 2006:467).

2.3. *Software* Pendukung

Dipakai untuk membantu peneliti dalam membangun sistem yang nanti akan menjadi sebuah aplikasi. Aplikasi pendukung untuk merancang sistem ini berupa:

2.3.1 UML (*Unified Modeling Language*)

Unified Modeling Language (UML) adalah standar yang sering digunakan untuk mendefinisikan persyaratan. Membuat analis serta desain, dan

menggambarkan arsitekstur dalam *Object Oriented Programming* (OOP) (Rosa & Shalahuddin, 2011).

Berikut adalah beberapa jenis diagram UML:

2.3.1.1. Use Case Diagram

Use case Diagaram adalah gambaran graphical dari *actor*, *use case*, dan interaksi yang memperkenalkan suatu sistem. *Use Case* mengambarkan hubungan dari satu atau lebih actor dengan sistem informasi yang mau dibuat. *Use Case* digunakan untuk mencari tahu fungsi apa yang ada dalam sistem informasi dan siapa yang berhak memanfaatkan fungsi tersebut.

Use Case Diagaram adalah permodelan untuk kelakuan sistem informasi yang mau dibuat, *Use Case* menggambarkan interaksi antara satu atau beberapa actor dengan sistem informasi yang mau dibuat. *Use Case* biasanya digunakan untuk mencari informasi tentang fungsi yang berada dalam sistem informasi dan siapa individu yang berhak memanfaatkan fungsi-fungsi tersebut. Ada dua poin utama pada *Use Case*, Yaitu Aktor dan *Use Case* (Rosa & Shalahuddin, 2015).

Tabel 2.2 Simbol Use Case

SIMBOL	DESKRIPSI
Use Case	Fungsi disediakan di sistem sebagai unit yang bertukar pesan antara unit atau actor, biasanya diekspresikan dengan menggunakan kata kerja di mulainya frase *use case*

Aktor / *Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem
Asosiasi/ *association*	Komunikasi antara kator dan *Use Case* yang berpartisipasi dalam *Use Case*
Ekstensi / *Extension*	Hubungan *use case* tambahan dengan *use case* dimana *case* yang ditambahkan dapat berdiri sendiri bahkan tanpa *use case* tambahan
Generalisasi / *Generalization* 	Generelasi hubungan dan spesialisasi antara dua *use case* dimana satu fungsi lebih umum dari pada yang lain.
Menggunakan / *Include*	Hubungan *use casei* tambahan dengan *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau syarat untuk memakai *use case* ini

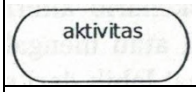
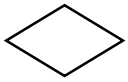
2.3.1.2. Activity Diagaram

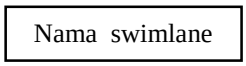
Activity Diagaram visualkan alur kerja, yang perlu dipertimbangkan di sini berarti bahwa diagram yang mengambarkan aktivitas sistem bukanlah apa yang dilakukan actor, sehingga aktivitas dapat dilakukan oleh sistem.

Activity Diagram mengambarkan *workflow* atau aktivitas sistem atau proses bisnis atau menu yang ada di dalam *software*. Yang perlu telitikan di sini adalah *activity diagram* yang menggambarkan aktivitas sitem bukanlah yang dilakukan actor (Rosa & Shalahuddin, 2015). *Activity Diagram* juga sering dipakai untuk menentukan hal-hal berikut:

a. Desain proses bisnis di mana setiap urutan kegiatan yang dijelaskan adalah sistem proses bisnis yang ditentukan
b. Urutan atau pengelompokan tampilan antarmuka sistem di mana setiap aktivitas dianggap memiliki desain antarmuka layar
c. Desain tes di mana setiap kegiatan dianggap membutuhkan tes yang perlu didefinisikan sebagai kasus uji
d. Design menu ditampilkan oleh perangkat lunak

Tabel 2.3 Simbol Activity Diagram

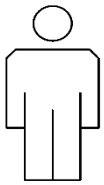
SIMBOL	DESKRIPSI
STATUS AWAL 	Status awal aktivitas sistem, setiap diagram aktivitas pasti memiliki status awal
Aktivitas / *Activity* 	Aktivitas dalam sistem, biasanya diawali dengan kata kerja
Percabangan / *Decision* 	Asosiasi percabangan dimana jika ada pemilihan

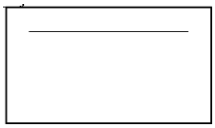
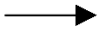
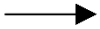
Penggabungan / *Join* 	Asosiasi penggabungan dimana beberapa aktivitas gabung menjadi satu
Status Akhir 	Status akhir akhir yang dilakukan sistem, setiap diagram aktivitas pasti memiliki status akhir
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

2.3.1.3. Sequence Diagram

Sequence Diagram merupakan gambungan dari *Diagram Class* dan *diagram Object* yang memiliki suatu gambaran model statis, ada juga yang bersifat dinamis, seperti Diagram Interaction.Diagram sequence merupakan salah satu diagram Interaction yangmenjelaskan bagaimana suatu operasi itu dilakukan; message (pesan) apa yang dikirimdan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu. Objek-objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut (Rosa & Shalahuddin, 2015).

Tabel 2.4 Simbol *Sequence Diagram*

SIMBOL	DESKRIPSI
Aktor / Actor 	Menggambarkan sesorang yang beriteraksi dengan sistem, di mana hanya bisa menginputkan informasi dan menerima indormasi dari sistem dan tidak memegang kendali pada use case
Garis hidup / lifeline 	Menyatakan *lifetime* satu objek

Objek 	Menyatakan objek yang berinteraksi pesan
Waktu Aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe *create* <<*create*>>	Menyatakan suatu objek membuat objek yang lain, arah panah mengaruh pada objek yang dibuat
Pesan tipe *call* Nama_metode() 	Memanggil operasi / metode yang ada
Pesam tipe *send* 	Mengirim informasi ke objek lainnya
Pesan tipe *return*	Kembalikan satu data / informasi setelah telah menjalankan suatu operasi / metode
Pesan tipe *destroy*	Mengakhiri hidup objek yang lain

2.3.1.4. Class Diagram

Class diagram adalah model statis yang menggambarkan struktur dan deskripsi class serta hubungannya antara class. Class diagram mirip ER-Diagram

pada perancangan database, bedanya pada ER-diagram tdk terdapat operasi/methode tapi hanya atribut (Rosa & Shalahuddin, 2015).

Kelas-kelas dalam struktur sistem harus bisa meneksekusi fungsi sebagai persyaratan sistem sehingga *developer* dapat membuat kelas dalam programnya seperti desain diagram kelas. Struktur-struktur kelas yang baik dalam diagram kelas harus memiliki jeni kelas-kelas berikut (Rosa & Shalahuddin, 2015):

1. *Main Class*

   Kelas yang memiliki fungsi awal ketika dibuka

2. *View* Class

   Kelas yang mengatur tampilan ke aktor.

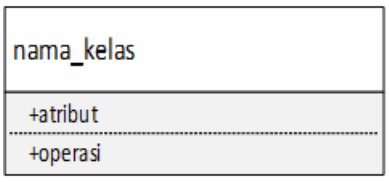
3. *Controller Class*

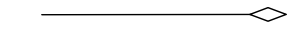
   Kelas yang memproses fungsi-fungsi yang akan diambil dari definisi *use case*

4. *Model Class*

   Kelas yang dipakai untuk memgang data menjadi suatu kesatuan yang diambil ataupun disimpan ke basis data

Tabel 2.5 Simbol *Class Diagram*

SIMBOL	DESKRIPSI
Kelas / *Class* 	Kelas pada struktur sistem

Antarmuka / *Interface* 	Tampilan / desain yang di tampilkan ke pemakai
Asosiasi berarah / *direct association* 	Relasi antarkelas dengan makna umum
Generalisasi / *Generalization*	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum-khusu)
Gantungan / *Dependency* 	Relasi antar kelas dengan makna keberagantungan kelas
Agresi / *aggregation* 	Relasi antar kelas dengan makna semua bagian

Biasanya untuk mengambarkan diagram-diagram (seperti diagram-diagram di atas), bisa mengunakan aplikasi yang bernama StarUML. StarUML adalah *platform* pemmodelan *software* yang mendukung UML (*Unified Modeling Language*). StarUML berdasarkan UML versi 1.4, menyiapkan 11 jenis *diagram*, serta kompatibel dengan notasi UUM 2.0 StarUML dan juga secara aktif mendukung pendekatan MDA (*Model Driven Architercture*). StarUML lebih dikenal untuk

Gambar 2.3 Logo StarUML

keunggulannya dalam hal menyesuaikan lapangan kerja pengguna, dan memiliki ekstensi tinggi untuk fungsinya (Triandini & Suardika, 2012: 1).

2.3.2 Visual Studio

Adalah sebuah perangkat lunak lengkap yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi console, aplikasi Windows, ataupun aplikasi Web.

Dengan Visual Studio ini, penulis menuliskan *coding* aplikasi memanfaatkan bahasa pemograman sebagai berikut.

Gambar 2.4 *Visual Studio*

a. Bahasa Pemograman Vb. Net

Adalah sebuah alat untuk mengembangkan dan membangun aplikasi yang bergerak di atas sistem .NET Framework, dengan menggunakan bahasa BASIC. Dengan menggunakan alat ini, para *programmer* dapat membangun aplikasi Windows Forms, Aplikasi web berbasis ASP.NET, dan juga aplikasi *command-line*. Alat ini dapat diperoleh secara terpisah dari beberapa produk lainnya (seperti *Microsoft Visual* C++, *Visual* C#, atau *Visual* J#), atau juga dapat diperoleh secara terpadu dalam Microsoft Visual Studio .NET.

2.3.3 Database SQL *Server*

Microsoft SQL Server adalah *software* untuk *manage* relasional basis data Microsoft (RDBMS). Bahasa dari query utama adalah Bahasa Transact - SQL, adalah penerapan SQL standar ANSI / ISO yang dipakai oleh *Microsoft* serta *Sybase*. SQL *server* bisa digunakan untuk pribadi maupun industri kecil, menengah, ataupun besar.

Gambar 2.5 Logo *SQL Server*

2.4. Penelitian Terdahulu

Penelitian ini dilakukan untuk meningkatkan hasil penelitian terdahulu yang sudah pernah diteliti, ada beberapa penelitian sebelumnya sebagai berikut:

Menurut Supyani, Bebas Widata, Wawan Laksito dengan judul "**APLIKASI DETEKSI KERUSAKAN MESIN SEPEDA MOTOR BEBEK 4 TAK DENGAN METODE *FORWARD CHAINING* (ISSN: 2338-4018, 2014)**", Pesatnya pertumbuhan sepeda motor tentunya juga harus didukung oleh kesiapan mekaniknya, sebab semakin banyak jumlah sepeda motor yang digunakan individu akan semakin banyak pula timbulnya kerusakan mesin. Dalam prakteknya pabrik telah melatih mekaniknya untuk menyelesaikan masalah kerusakan sepeda motor yang muncul, karena banyaknya pengguna sepeda motor maka jumlah itu tidak cukup untuk menyelesaiakan masalah kerusakan sepeda motor, untuk itu banyak

bengkel yang berdiri untuk dapat membantu menyelesaikan masalah kerusakan sepeda motor.

Menurut Kresna Amanda dengan judul "**PENERAPAN SISTEM PAKAR UNTUK MENDETEKSI PENYAKIT PADA KEHAMILAN (ISSN: 1978-1946, 2015)**", Kehamilan adalah suatu fenomena fisiologis yang dimulai dengan pembuahan dan diakhiri dengan proses persalinan. Selama kehamilan, ibu dan janin adalah unit fungsi yang tak terpisahkan. Meskipun terlihat dengan kondisi kehamilan yang sehat bukan berarti ibu dan janin dalam keadaan baik – baik saja. Namun kurangnya informasi atau sosialisasi tentang penyakit kehamilan akan menyebabkan mereka baru mengetahui adanya penyakit yang menyertai kehamilannya setelah stadium lanjut.

Menurut Rosmawati Tamin dengan judul "**SISTEM PAKAR UNTUK DETEKSI KERUSAKAN PADA PRINTER MEMANFAATKAN METODE *FORWARD CHAINING* (SSN: 2442–4512, 2015)**", Mengganti pengetahuan manusia ke dalam bentuk sistem dengan mengakomodasi kemampuan / keahlian Seorang ahli untuk mengolah analisis suatu masalah sehingga sistem tersebut dapat bekerja dengan sistem untuk memecahkan masalah seperti halnya manusia melakukannya dan memecahkan masalah. Kerusakan printer terkadang merupakan masalah besar ketika orang awam tidak tahu lokasi kesalahan printer, itu membutuhkan sistem yang mampu bekerja secara otomatis untuk memberikan solusi untuk merusak printer. Penelitian ini bertujuan untuk merancang sistem yang dapat digunakan untuk menangani kerusakan printer. Pengguna aplikasi ini tampaknya berhadapan langsung dengan para ahli di aspek perangkat keras,

terutama printer. Perencanaan sistem dilakukan dengan membuat basis pengetahuan menggunakan pohon keputusan dan jika-maka aturan sebagai representasi pengetahuan. Sistem ini dibuat dengan menggunakan metode forward chaining dan bahasa pemrograman Visual Basic. Hasil penelitian ini mengungkapkan jenis kerusakan yang terjadi pada printer dan penanganan kerusakan. Pengujian aplikasi juga dilakukan untuk menentukan akurasi dan variasi serta keramahan pengguna dan fleksibilitas sistem. Hasil dari keseluruhan pengujian ini dapat disimpulkan bahwa program ini cukup baik walaupun jenis kerusakan yang dihasilkan tidak lengkap karena sistem ini hanya mendeteksi 15 jenis kerusakan mesin secara umum.

Menurut Samy Salim Abu Nazer dan Mohammed Ibrahim Alhabbash dengan Judul "***MALE INFERTILITY EXPERT SYSTEM DIAGNOSES AND TREATMENT (ISSN : 2429-5396, 2016)****", Infertility affects up to 15% of couples of reproductive age all over the world, the prevalence of infertility is said to be increasing globally and Male infertility is the most prevalent. Infertility means no pregnancy after one year of marriage, 40% due to male factor and 40% due to female factor and 20% due to both factors. Many men do not know something about infertility.*

Menurut Samy Saling Abu Nazer dan Ali Osama Mahdi dengan judul "***A PROPOSED EXPERT SYSTEM FOR FOOT DISEASES DIAGNOSIS (ISSN : 2429-5396, 2008)***", *Feet are complex structures with 26 bones, 33 joints, and many muscles, nerves, ligaments of various types. Any part of the foot may be affected. Some leg disorders may arise only with blunt pain, but on the other hand other leg*

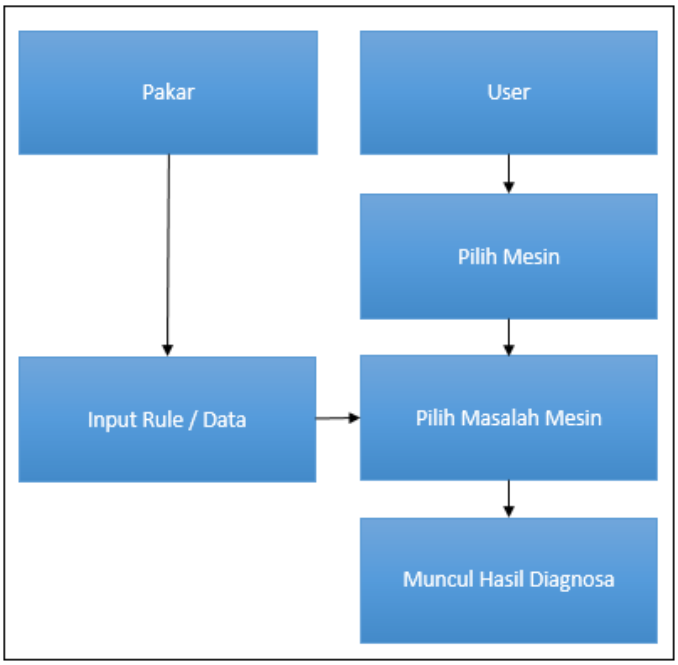
disorders can be very severe and limit the ability to walk or tolerate weight. The bulk of small foot pain cases are treated at home with special care. When severe pain is encountered, it requires several types of medical treatment. If proper treatment for leg pain is not done quickly, it can cause damage to the foot. Objective: The main objective of this expert system is to get the right diagnosis of the disease and the right treatment. Method: In this paper the design of the proposed Expert System is produced to help Podiatric doctors in diagnosing many foot diseases such as clubfoot, toes, stress fractures, ankle or ankle fractures, sprained legs, turtle legs, turtle feet, flat feet , Plantar Fasciitis, Warts, Bunion, Rheumatoid Arthritis, Gout, Heel Spurs, Athlete's Foot, Morton Neuroma, Cellulitis, Frostbite, and Gangrene. The proposed expert system provides an overview of the foot disease given, the cause of the disease described and treatment of the disease if possible is given. SL5 Language The Object Expert System is used to design and implement the proposed expert system. Results: The proposed expert system for diagnosing foot disease was evaluated by medical students and they were satisfied with their performance. Conclusion: The proposed expert system is very useful for Podiatric doctors, patients with foot problems and newly graduated doctors.

2.5. Kerangka Pemikiran

Identifikasi masalah dalam penelitian ini berupa banyaknya *operator* manufaktur yang tidak bisa memperbaiki mesin diri sendiri karena kurangnya

pengetahuan dan tidak ada sebuah sistem yang bisa membantu pengguna untuk mengecek masalah pada mesin manufaktur, waktu untuk menunggu kedatangan teknisi dapat menyebabkan tidak capainya target hasil produksi karena *operator* tidak dapat bekerja ketika mesin sedang ada masalah.

Kerangka pemikiran sebagai berikut:



Gambar 2.6 Kerangka Pemikiran Penelitian

Uraian dari poin-poin diatas adalah sebagai berikut:

2.5.1 Pakar / Ahli

Pemakai yang memiliki akses untuk seluruh halaman yang terdapat pada sistem.

2.5.2 *User*

Pengguna system atau *operator*, biasanya mengunakan sistem untuk melaporkan kerusakan pada mesin ke dalam sistem. Tidak mempunya autoritas untuk mengisi atau menambah *rule* ataupun *user* lain.

2.5.3 Input *Rule* / Data

Tambahkan data *Rule* dan mesin ke sistem, sehingga Pengguna dapat laporkan mesin yang rusak.

2.5.4 Pilih Mesin

Pengguna/User memilih Mesin yang ingin di Deteksi.

2.5.5 Input Masalah Mesin

Apabila *operator* mengalami masalah saat sedang menggunakan mesin, maka dia daftar masalah tersebut ke dalam sistem, serta gejala-gejala yang terjadi.

2.5.6 Hasil Deteksi

Dari gejala-gejala yang telah diinput, sistem Akan menampilkan hasil deteksi kerusakan mesin.