BAB II

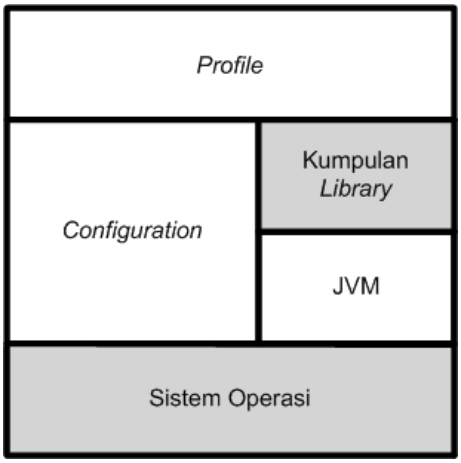
LANDASAN TEORI

2.1 *Java 2 Micro Edition* (J2ME)

2.1.1 Pengertian J2ME

Java2 Micro Edition atau yang biasa disebut J2ME adalah lingkungan pengembangan yang didesain untuk meletakkan perangkat lunak Java pada barang elektronik beserta perangkat pendukungnya. Pada J2ME, jika perangkat lunak berfungsi baik pada sebuah perangkat maka belum tentu juga berfungsi baik pada perangkat yang lainnya. J2ME membawa Java ke dunia informasi, komunikasi, dan perangkat komputasi selain perangkat komputer desktop yang biasanya lebih kecil dibandingkan perangkat komputer desktop. J2ME biasa digunakan pada telepon selular, pager, *Personal Digital Assistants* (PDA) dan sejenisnya.

J2ME adalah bagian dari J2SE, karena itu tidak semua *library* yang ada pada J2SE dapat digunakan pada J2ME. Tetapi J2ME mempunya beberapa *library* khusus yang tidak dimiliki J2SE. Arsitektur J2ME dapat dilihat pada gambar berikut :

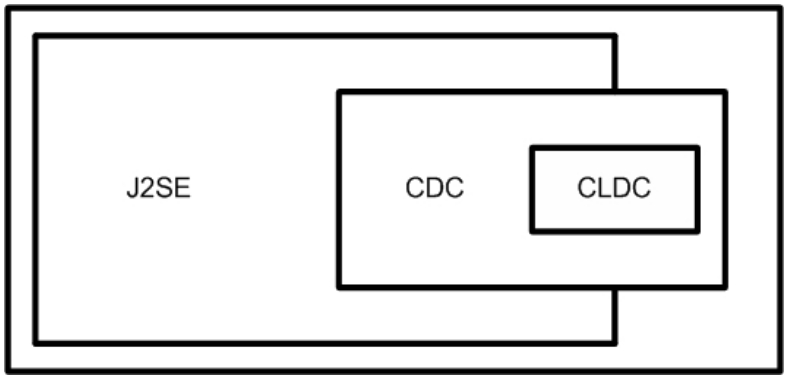


Gambar 2.1 Arsitektur J2ME

Teknologi J2ME juga memiliki beberapa keterbatasan, terutama jika diaplikasikan pada ponsel. J2ME sangat tergantung pada perangkat (*device*) yang digunakan, bisa dari segi merk ponsel, maupun kemampuan ponsel dan dukungannya terhadap teknologi J2ME. Misalnya, jika sebuah ponsel tidak memiliki kamera, maka jelas J2ME pada ponsel tersebut tidak dapat mengakses kamera. Keterbatasan lainnya adalah pada ukuran aplikasi, karena memori pada ponsel sangat terbatas. Sebagian besar ponsel tidak mengijinkan aplikasi J2ME menulis pada file karena alasan keamanan.

Configuration merupakan *Java library minimum* dan kapabilitas yang dipunya oleh para pengembang J2ME, yang maksudnya sebuah *mobile device* dengan kemampuan *Java* akan dioptimalkan untuk menjadi sesuai. *Configuration* hanyalah mengatur hal-hal tentang kesamaan sehingga dapat dijadikan ukuran kesesuaian antar *device*. Misalnya sebuah lampu sepeda dirancang sedemikian sehingga dapat digunakan oleh berjenis-jenis sepeda. Dalam J2ME telah didefinisikan dua buah konfigurasi yaitu CLDC (*Connected Limites Device*

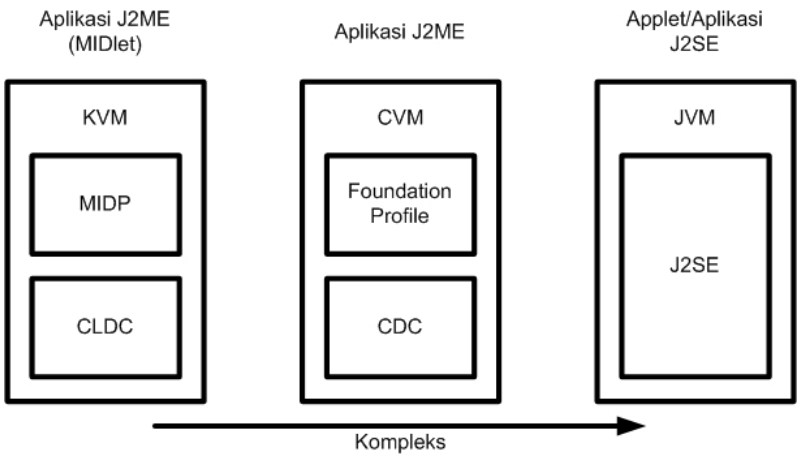
Configuration) untuk perangkat kecil dan CDC (*Connected Device Configuration*) untuk perangkat yang lebih besar. Lingkup CLDC dan CDC dapat dilihat pada gambar di bawah ini :



Gambar 2.2 Lingkup Configuration

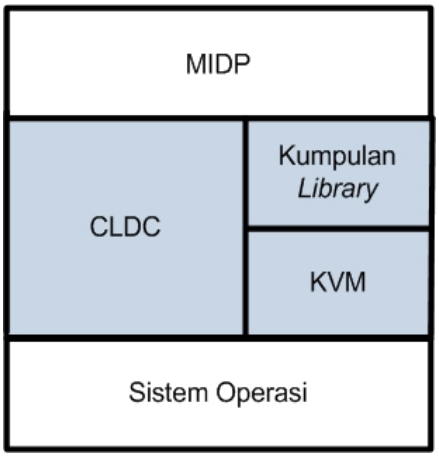
Profile berbeda dengan *configuration*, *profile* membahas sesuatu yang spesifik untuk sebuah perangkat. Sebagai contoh misalnya, sebuah sepeda dengan merk tertentu, tentunya mempunyai ciri spesifik dengan sepeda lain.

Dalam gambar J2ME terdapat dua buah *profile* yaitu MIDP dan *Foundation Profile.* Keterhubungan antara *configuration* dan *profile* yang ada pada J2ME beserta jenis mesin virtualnya dapat dilihat pada gambar berikut :



Gambar 2.3 Hubungan J2ME dan J2SE

2.1.2 *Connected Limited Device Configuration* (CLDC)



Gambar 2.4 Arsitektur CLDC

CLDC atau *Connected Limited Device Configuration* adalah perangkat dasar dari J2ME, spesifikasi dasar yang berupa *library* dan API yang diimplemetasikan pada J2ME, seperti yang digunakan pada telepon selular, pager dan PDA. Perangkat tersebut dibatasi dengan keterbatasan memori, sumber daya dan kemampuan memproses. Spesifikasi CLDC pada J2ME adalah spesifikasi minimal dari *package*, kelas dan sebagian fungsi *Java Virtual Machine* yang dikurangi agar dapat diimplementasikan dengan keterbatasan sumber daya pada alat-alat tersebut, JVM yang digunakan disebut KVM (*Kilobyte Virtual Machine*). Posisi CLDC pada arsitektur J2ME dapat dilihat pada gambar di atas.

2.1.3 *Connected Device Configuration* (CDC)

CDC atau *Connected Device Configuration* adalah spesifikasi dari konfigurasi J2ME. CDC merupakan komunitas proses pada *Java* yang mempunyai standardisasi. CDC terdiri dari virtual machine dan kumpulan *library* dasar untuk dipergunakan pada profile industri. Implementasi CDC pada J2ME

adalah *source code* yang menyediakan sambungan dengan macam-macam *platform*.

Perbedaan antara CDC dan CLDC dapat dilihat dari tabel dibawah ini

Tabel 2.1 Tabel Perbandingan Antara CDC dan CLDC

CLDC	CDC
Mengimplementasikan subset dari J2SE	Mengimplementasikan seluruh fitur pada J2SE
JVM yang digunakan dikenal dengan nama KVM	JVM yang digunakan dikenal dengan nama CVM
Digunakan pada perangkat handled dengan ukuran memori terbatas (160 – 512 Kbytes)	Digunakan pada perangkat handled dengan ukuran memori minimal 2 Mbytes
Prosesor : 16 Bit atau 32 Bit	Prosesor : 32 Bit

2.2 Profil

Profil merupakan bagian perluasan dari konfigurasi, yaitu sekumpulan kelas yang terdapat pada konfigurasi, terdapat juga kelas-kelas spesifik yang didefinisikan lagi di dalam profil. Dengan kata lain, profil akan membantu secara fungsional yaitu dengan menyediakan kelas-kelas yang tidak terdapat di level konfigurasi. Berikut ini adalah profil J2ME yang tersedia saat ini yaitu :

1. MIDP (*Mobile Information Device Profile*)

   MIDP adalah profil yang disediakan oleh Sun Microsystems. MIDP menyediakan librari-librari Java untuk implementasi dasar antarmuka (GUI), implementasi jaringan (*networking*), database, dan timer. MIDP dirancang khususnya untuk *wireless phone* dan *pager.*

2. PDAP (*Personal Digital Assistant Profile*)

   Yaitu profil untuk PDA yang memperluas fungsi-fungsi pada konfigurasi CLDC dan digunakan khusus untuk menambahkan kemampuan-kemampuan lebih apabila dibandingkan dengan penggunaan profil MIDP.

3. *Foundation Profile*

   Yaitu profil yang digunakan untuk konfigurasi CDC. Profil ini menambahkan beberapa kelas dari J2SE ke dalam konfigurasi CDC, dan berperan juga sebagai pondasi untuk membentuk profil baru lainnya.

4. *Personal Profile*

   Yaitu profil yang mendefinisikan ulang PersonalJava sebagai profil yang dapat digunakan sebagai profil dalam J2ME. Profil ini merupakan hasil perluasan dari *Foundation Profile.*

5. *RMI Profile*

   Yaitu profil yang menambahkan dukungan RMI (*Remote Method Invocation*) ke dalam konfigurasi CDC.

2.2.1 *Mobile Information Device Profile* (MIDP)

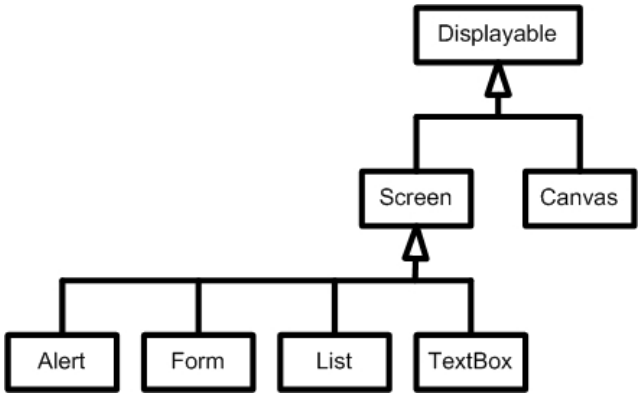
MIDP atau *Mobile Information Device Profile* adalah spesifikasi untuk sebuah profil J2ME. MIDP memiliki lapisan di atas CLDC, API tambahan untuk daur hidup aplikasi, antarmuka, jaringan dan penyimpanan persisten. Pada saat ini terdapat MIDP 1.0 dan MIDP 2.0. Fitur tambahan MIDP 2.0 dibanding MIDP 1.0 adalah API untuk multimedia. Pada MIDP 2,0 terdapat dukungan memainkan *tone*, *tone sequence* dan file WAV walaupun tanpa adanya *Mobile*

Media API (MMAPI). Posisi MIDP pada arsitektur J2ME dapat dilihat pada gambar di atas. Berikut adalah perbandingan MIDP 1.0 dan MIDP 2.0 :

Tabel 2.2 Perbandingan MIDP 1.0 dan MIDP 2.0

Spesifikasi	MIDP 1.0	MIDP 2.0
Display	96 x 54	96 x 54
Kedalaman Display	1 -bit	1 -bit
Bentuk piksel (rasio aspek)	Mendekati 1:1	Medekati 1:1
Input	Keyboard dan touch screen	Keyboard dan touch screen
Memori	128 KB memori non-volatile untuk komponen MIDP. 8 KB memori non-volatile untuk data persistence yang dibuat oleh aplikasi. 32 KB memori volatile untuk JRE.	256 KB memori non-volatile untuk komponen MIDP. 8 KB memori non-volatile untuk data persistence yang dibuat oleh aplikasi. 128 KB memori volatile untuk JRE.
Jaringan	Dua arah, tanpa kabel (wireless)	Dua arah, tanpa kebel (wireless)
Library J2ME yang bukan merupakan *library* J2SE	javax.microedition.lcdui, javax.microedition.midlet, javax.microedition.rms	javax.microedition.lcdui, javax.microedition.midlet, javax.microedition.rms, javax.microedition.lcdui.game, javax.microedition.media, javax.microedition.pki
Multimedia		Memiliki kemampuan untuk memainkan file multimedia (suara dan video)

MIDP *User Interface* API memiliki API level tinggi dan level rendah. API level rendah berbasiskan penggunaan dari kelas abstrak *Canvas,* sedangkan kelas API level tinggi antara lain *Alert, Form, List* dan TextBox yang merupakan eksistensi dari kelas abstrak *Screen*. API level rendah lebih memberikan kemudahan kepada pengembang untuk memodifikasi sesuatu dengan kehendaknya, sedangkan API level tinggi biasanya hanya memberikan pengaksesan yang terbatas. Arsitektur antarmuka MIDP dapat dilihat pada gambar di bawah ini:



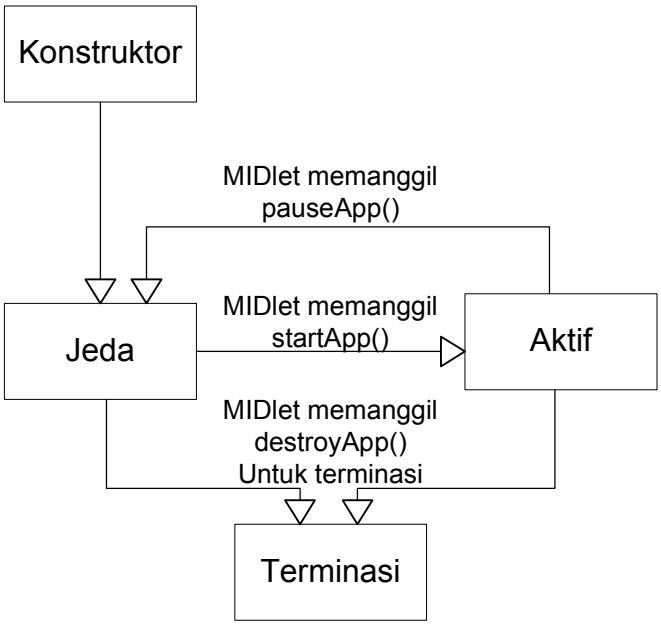
Gambar 2.5 MIDP User Interface

2.3 MIDlet

MIDlet adalah bagian dari kelas *javax.microedition.midlet.MIDlet* yang didefinisikan pada MIDP. MIDlet berupa sebuah kelas abstrak yang merupakan sub kelas dari bentuk dasar aplikasi sehingga antarmuka antara aplikasi J2ME dan aplikasi manajemen pada perangkat dapat terbentuk.

2.3.1 Daur Hidup MIDlet

MIDlet terdiri dari beberapa metode yang harus ada, yaitu *construktor() protected void startApp() throws MIDletStateChangeException, protected void pauseApp(), protected void destroyApp(boolean unconditional) throws MIDletStateChangeException.* Alur hidup MIDlet dapat dilihat pada gambar 2.6.



Gambar 2.6 Alur Hidup MIDlet

Ketika MIDlet dijalankan maka akan diinisialisasi dengan kondisi *pause* dan dijalankan *pauseApp()*, kondisi berikutnya adalah fungsi MIDlet dijalankan, yaitu pada *startApp().* Metode tersebut diimplementasikan sebagai *protected* dengan maksud agar MIDlet lain tidak dapat memanggil metode tersebut.

Ketika keluar dari MIDlet, maka metode *destroyApp()* akan dijalankan sebelum MIDlet benar-benar tidak berjalan lagi. *DestroyApp()* akan memanggil *notifyDestroyed(),* dan *notifyDestroyed()* akan memberitahu platform untuk menterminasi MIDlet dan membersihkan semua sumber daya yang mengacu pada MIDlet.

Dalam implementasinya, MIDlet memiliki struktur direktori antara lain sebagai berikut :

1. *src*

   menyimpan *source code* untuk MIDlet dan kelas lain yang diperlukan.

2. *res*

   Menyimpan sumber daya yang dibutuhkan oleh MIDlet, seperti misalnya gambar icon.

3. *lib*

   menyimpan file JAR atau ZIP yang berisi *library* tambahan yang dibutuhkan MIDlet.

4. *bin*

   menyimpan file JAR, JAD dan file *manifest* yang berisi muatan komponen MIDlet.

2.4 UML (*Unified Modelling Language*)

Unified Modelling Language (UML) adalah sebuah bahasa untuk menentukan, visualisasi, konstruksi, dan mendokumentasikan *artifacts* dari sistem *software*, untuk memodelkan bisnis, dan sistem *nonsoftware* lainnya. *Artifacts* adalah sepotong informasi yang digunakan atau dihasilkan dalam suatu proses rekayasa *software*. *Artifacts* dapat berupa model, deskripsi, atau software.

Untuk membuat suatu model, UML memiliki diagram grafis yang diberi nama berdasarkan sudut pandang yang berbeda-beda terhadap sistem dalam proses analisa atau rekayasa. Diagram grafis tersebut antara lain :

1. *Use case diagram*
2. *Class diagram*

3. *Behavior diagram :*
    a. *Statechart diagram*
    b. *Activity diagram*
    c. *Interaction diagram :*
        1. *Sequence diagram*
        2. *Collaboration diagram*
4. *Implementation diagram :*
    a. *Component diagram*
    b. *Deployment diagram*

2.4.1 *Use Case Diagram*

Use case diagram menjelaskan manfaat sistem jika dilihat menurut pandangan orang yang berada diluar sistem (*actor*). Diagram ini menunjukkan fungsionalitas suatu sistem atau kelas dan bagaimana sistem berinteraksi dengan dunia luar.

2.4.2 *Class Diagram*

Class diagram membantu dalam visualisasi struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak dipakai. *Class diagram* memperlihatkan hubungan antarkelas dan penjelasan detail tiap-tiap kelas didalam model desain dari suatu sistem .

Selama proses analisis, *class diagram* memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. Selama tahap desain,

class diagram berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat.

2.4.3 *Statechart Diagram*

Statechart diagram dapat digunakan untuk memodelkan perilaku dinamis satu kelas atau objek. *Statechart diagram* juga dapat memperlihatkan urutan keadaan sesaat (*state*) yang dilalui sebuah objek, kejadian yang menyebabkan sebuah transisi dari satu state atau aktivitas kepada yang lainnya, dan aksi yang menyebabkan perubahan satu *state* atau aktivitas.

2.4.4 *Activity Diagram*

Activity diagram memodelkan alur kerja sebuah proses bisnis dan urutan aktivitas dalam suatu proses. Diagram ini sangat mirip dengan *flowchart* karena dapat memodelkan sebuah alur kerja satu aktivitas ke aktivitas lainnya atau dari satu aktivitas ke dalam keadaan sesaat (*state*). Seringkali bermanfaat bila membuat sebuah proses *activity diagram* terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan.

2.4.5 *Sequence Diagram*

Sequence diagram menjelaskan interaksi objek yang disusun dalam suatu urutan waktu. Diagram ini secara khusus berasosiasi dengan *use case*. *Sequence diagram* memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk menghasilkan sesuatu didalam *use case*.

2.4.6 *Collaboration Diagram*

Collaboration diagram melihat pada interaksi dan hubungan terstruktur antarobjek. Tipe diagram ini menekankan pada hubungan antarobjek, berbeda dengan *sequence diagram* yang menekankan pada urutan kejadian. *Collaboration diagram* digunakan sebagai alat untuk menggambarkan interaksi yang mengungkapkan keputusan mengenai perilaku sistem.

2.4.7 *Component Diagram*

Component diagram menggambarkan alokasi semua kelas dan objek ke dalam komponen-komponen dalam desain fisik sistem software. Diagram ini memperlihatkan pengaturan dan kebergantungan antara komponen-komponen software, seperti *source code, binary code*, dan komponen tereksekusi (*executable components*).

2.4.8 *Deployment Diagram*

Deployment diagram memperlihatkan pemetaan software kepada hardware. Setiap model hanya memiliki satu *diagram deployment*.

2.4.9 Tujuan Perancangan UML

Perancangan UML mempunya beberapa tujuan utama. Tujuan-tujuan tersebut adalah sebagai berikut :

1. Menyediakan bahasa pemodelan visual yang ekspresif dan siap pakai untuk pengembangan dan pertukaran model-model yang berarti.

2. Menyediakan mekanisme perluasan dan spesifikasi untuk memperluas konsep inti.
3. Mendukung spesifikasi independen pada bahasa pemrograman dan proses pengembangan tertentu.
4. Menyediakan basis formal untuk bahasa pemodelan.
5. Mendukung konsep-konsep pengembangan level tinggi, seperti komponen, kolaborasi, *framework* dan *pattern*.

2.5 Kecerdasan Buatan

2.5.1 Pengertian Kecerdasan Buatan

Kecerdasan buatan adalah suatu ilmu yang mempelajari cara membuat komputer melakukan sesuatu seperti yang di lakukan oleh manusia. Kecerdasaan buatan (*Artificial Intelegence*) merupakan kawasan penelitian, aplikasi dan instruksi yang terkait dengan pemprograman komputer yang melakukan sesuatu hal dalam pandangan manusia adalah cerdas. Kecerdasan buatan (AI) sebagai sebuah studi tentang bagaimana membuat komputer melakukan hal-hal yang pada saat ini dapat dilakukan lebih baik oleh manusia.

Ada tiga tujuan kecerdasan buatan, yaitu membuat komputer lebih cerdas, mengerti tentang kecerdasan dan membuat mesin lebih berguna. Yang dimaksud kecerdasan adalah kemampuan untuk belajar atau mengerti dari pengalaman, memahami pesan yang kontradiktif dan ambigu, menanggapi dengan cepat dan baik atas situasi yang baru. Menggunakan penalaran dalam memecahkan masalah serta menyelesaikan dengan efektif.

2.5.2 Klasifikasi Program Kecerdasan Buatan

Makin pesatnya perkembangan teknologi menyebabkan adanya perkembangan dan perluasan lingkup yang membutuhkan kehadiran kecerdasan buatan, karakteristik cerdas mulai dibutuhkan diberbagai disiplin ilmu dan teknologi. Kecerdasan tidak hanya dominan dibidang ilmu komputer (informatika), namun juga sudah merambah di berbagai disiplin ilmu yang lain.

Enam golongan dasar dalam kecerdasan buatan adalah :

1. Sistem Pakar (*Expert System*)

   Disini komputer digunakan sebagai sarana untuk menyimpan pengetahuan para pakar. Dengan demikian komputer akan memiliki keahlian untuk menyelesaikan permasalahan dengan meniru keahlian yang dimiliki oleh pakar.
2. Pengolahan Bahasa ALami (*Natural Language Proccesing*) dan Peniruan Sematik (*Sematik Modeling*) Seperti penerjemah dari suatu bahasa kebahasa lain.
3. Pembuktian Teorema (*Theorem Proving*) dan Penalaran Terotomatisasi (*Automatic Reasoning*), Seperti program *Simon's Logic Theorist* yang bermanfaat untuk membuktikan suatu dalil matematis dari sejumlah aksioma.
4. Robotika (*Robotics*) dan perencanaan (*Planning*) seperti robot-robot dalam industri
5. Peniruan perilaku manusia (*Human Performance Modeling*), perangkat lunak yang dapat meniru perilaku manusia dewasa ini dipahami oleh para

psikolog untuk menganalisa teori *Freud* dan berbagai teori psikologi lainnya.

6. Permainan game cerdas (*Game Playing*), seperti permainan catur.

2.6 Sistem Pakar

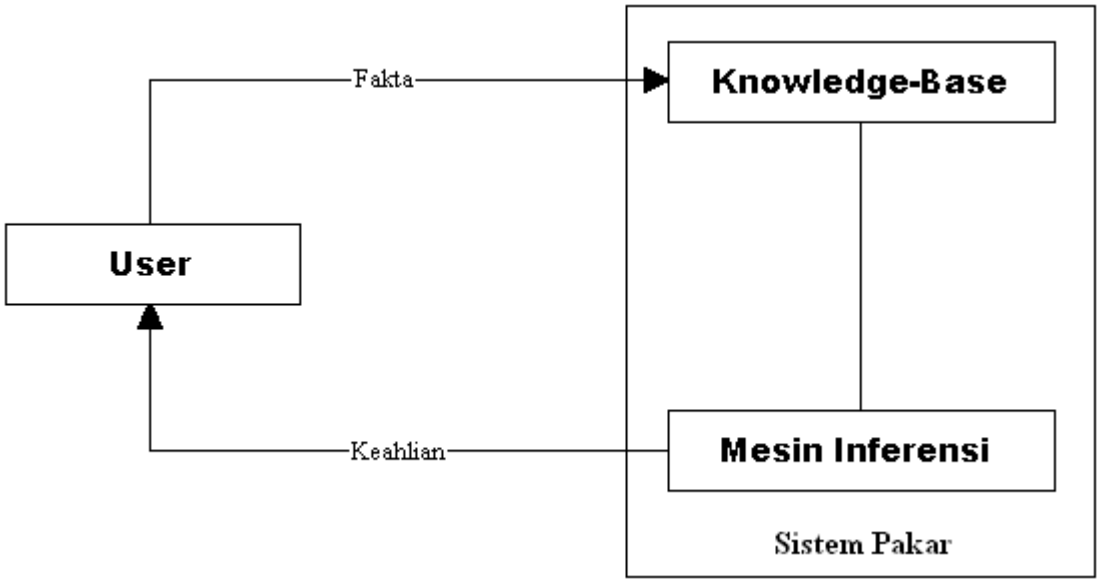
2.6.1 Pengertian Sistem Pakar

Sistem pakar adalah sistem berbasis komputer yang menggunakan pengetahuan, fakta dan teknik penalaran dalam memecahan masalah. Yang biasanya hanya dapat dipecahkan oleh seorang pakar dalam bidang tersebut.

Pada dasarnya sistem pakar diterapkan untuk mendukung aktivitas pemecahan masalah. beberapa pemecahan masalah yang dimaksud antara lain : pembuatan keputusan *(decicion making),* pemaduan pengetahuan *(knowledge fusing),* pembuatan desain *(designing),* perencanaan *(planning),* prakiraan *(forecasting),* pengaturan *(regulating),* pengendalian *(controlling),* diagnosis *(diagnosing),* perumusan *(prescribing)*, penjelasan *(explaining),* pemberian nasihat *(advising),* dan pelatihan *(tutoring).* Selain itu sistem pakar juga dapat berfungsi sebagai asisten yang pandai dari seorang pakar.

Sistem pakar dibuat pada wilayah pengetahuan tertentu untuk suatu kepakaran tertentu yang mendekati kemampuan manusia disalah satu bidang. Sistem pakar mencoba mencari solusi yang memuaskan sebagaimana yang dilakukan seorang pakar. Selain itu sistem pakar juga dapat memberikan penjelasan terhadap langkah yang diambil dan memberikan alasan atas saran atau kesimpulan yang ditemukan.

Biasanya sistem pakar hanya digunakan untuk memecahkan masalah yang sulit untuk dipecahkan dengan pemrograman biasa, mengingat biaya yang diperlukan untuk membuat sistem pakar jauh lebih besar dari pembuatan sistem biasa. Lebih jelas mengenai konsep dasar sistem pakar ini dapat dilihat pada gambar :



Gambar 2.7 Konsep Dasar Fungsi Sistem Pakar

2.6.2 Ciri – Ciri Sistem Pakar

Sistem pakar memiliki ciri- ciri sebagai berikut :

1. Terbatas pada bidang yang terpilih
2. Dapat memberikan penalaran untuk data – data yang tidak lengkap atau tidak pasti.
3. Dapat mengemukakan rangkaian alasan yang diberi hanya dengan cara yang dapat dipahami.
4. Berdasarkan pada *rule* atau kaidah tertentu

5. Dirancang untuk dapat dikembangkan secara bertahap.
6. Outputnya bersifat nasihat atau anjuran.
7. Output tergantung dari dialog dengan user
8. *Knowledge base* dan *inference engine* terpisah.

2.6.3 Keuntungan dan Kelemahan Sistem Pakar

a. Keuntungan Sistem Pakar :
    1) Memungkinkan orang awam dapat mengerjakan pekerjaan para ahli.
    2) Dapat melakukan proses secara berulang secara otomatis.
    3) Menyimpan pengetahuan dan keahlian para pakar.
    4) Meningkatkan output dan produktivitas.
    5) Meningkatkan kualitas.
    6) Memiliki kemampuan untuk mengakses pengetahuan.
    7) Meningkatkan kapabilitas dalam penyelesaian masalah.
    8) Menghemat waktu dalam pengambilan keputusan.
b. Kelemahan Sistem Pakar :
    1) Biaya mahal.
    2) Sulit dikembangkan.
    3) Hasil yang diharapkan tidak 100% benar.

2.6.3.1 Perbandingan Pakar dengan Sistem Pakar

No.	Faktor Pembanding	Pakar	Sistem Pakar
1	Kesediaan Waktu	Hari kerja/sesuai aturan yang disepakati	Setiap saat
2	Geografis	Lokal/tertentu	Dimana saja
3	Keamanan	Tidak terganti	Dapat diganti
4	Ketersediaan	Dapat habis	Tidak dapat habis
5	Performansi	Variable	Konsisten
6	Biaya	Tinggi	Terjangkau

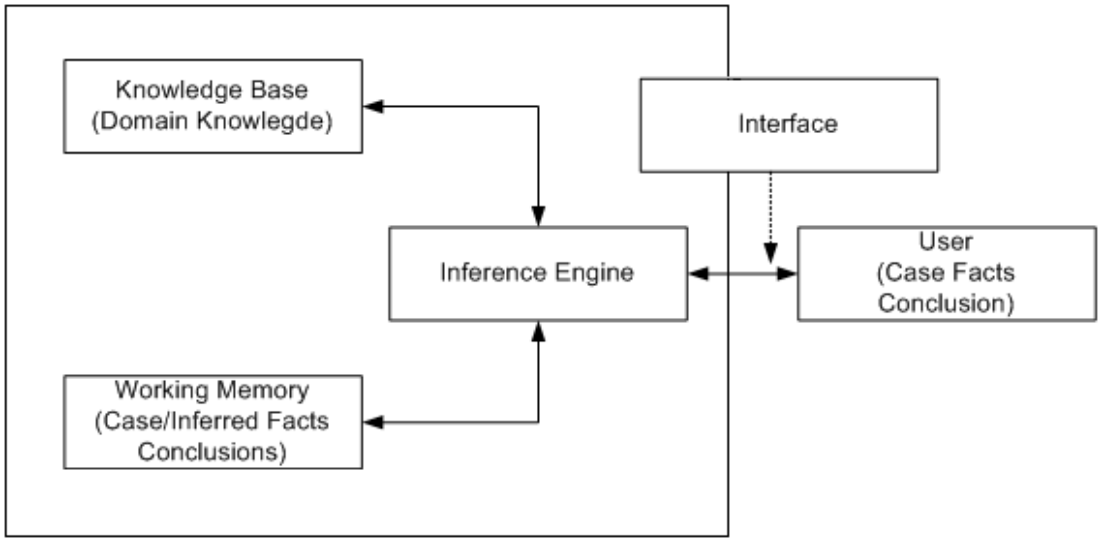
Tabel 2.3 Perbandingan antara pakar dengan sistem pakar.

2.6.4 Komponen Sistem Pakar

Sistem pakar disusun oleh dua bagian utama, yaitu :

1. Lingkungan pengembangan (*development environment*), yaitu digunakan untuk memasukkan pengetahuan pakar ke dalam lingkungan sistem pakar.
2. Lingkungan konsultasi (*consultation environment*), yaitu digunakan oleh pengguna yang bukan pakar guna memperoleh pengetahuan pakar.

Komponen-komponen sistem pakar dapat dilihat pada gambar :



Gambar 2.8 Komponen-komponen Sistem Pakar

a. *Knowledge Base*

Knowledge base (basis pengetahuan) adalah sebuah basis data yang menyimpan aturan-aturan tentang suatu *domain knowledge*/pengetahuan tertentu. *Knowledge base* merupakan inti dari program sistem pakar karena basis pengetahuan itu merupakan presentasi pengetahuan (*knowledge presentation*).

Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah. Komponen sistem pakar ini disusun atas dua elemen dasar, yaitu fakta dan aturan.

b. *Working Memory*

Working memory adalah bagian yang mengandung fakta-fakta, yaitu :

1. Fakta awal pada saat sistem beroperasi.
2. Fakta-fakta yang diperoleh pada saat sistem beroperasi.

Sistem akan mencocokkan fakta yang diperoleh dengan pengetahuan yang ada di basis pengetahuan untuk mencari fakta berikutnya ataupun untuk mendapatkan suatu kesimpulan.

c. *Inference Engine*

Inference engine (mesin inferensi) mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Mesin inferensi adalah program komputer yang memberikan metodologi untuk penalaran tentang informasi yang ada dalam basis pengetahuan dan dalam *workplace/working memory*, dan untuk memformulasikan kesimpulan.

Inference engine (mesin inferensi) adalah prosesor dalam sistem pakar yang bertugas untuk mencocokkan fakta yang diperoleh working memory dan domain knowledge base untuk mengahasilkan suatu kesimpulan. Mesin memulai pelacakannya dengan mecocokkan kaidah-kaidah dalam basis pengetahuan dengan fakta-fakta yang ada dalam basis data. Terdapat dua teknik inferensi, yaitu :

1. Penalaran maju (*forward chaining*), yaitu pencocokkan premise-premise yang ada untuk menghasilkan kesimpulan.
2. Penalaran mundur (*backward chaining*), yaitu mencari fakta berdasarkan kesimpulan untuk membuktikan bahwa kesimpulan adalah benar.

d. *Interface*

Interface (antarmuka) merupakan mekanisme yang digunakan oleh pengguna dan sistem pakar untuk berkomunikasi. Antarmuka menerima informasi dan pemakai dan mengubahnya ke dalam bentuk yang dapat diterima oleh sistem. Selain itu antarmuka menerima informasi dari sistem dan menyajikannya ke dalam bentuk yang dapat dimengerti oleh pemakai.

2.6.5 Representasi Pengetahuan

Menurut Turban, ada beberapa tipe pengetahuan yang bisa dikategorikan dalam bentuk keahlian, yaitu :

1. Teori-teori yang mendasari suatu permasalahan.
2. Aturan-aturan baku dan prosedur-prosedur yang berkaitan dengan permasalahan tertentu.
3. Aturan-aturan (*heuristik*) tentang apa yang harus dikerjakan dalam suatu permasalahan yang diberikan.
4. Strategi-strategi global untuk pemecahan dari tipe-tipe ini.
5. *Meta knowledge* (pengetahuan dari pengetahuan).
6. Fakta atau bukti tentang suatu permasalahan.

Dalam sistem pakar, pengetahuan yang telah diuraikan direpesentasikan ke dalam bentuk yang dapat diproses oleh komputer. Ada empat jenis representasi pengetahuan, yaitu :

1. Logika
2. Jaringan Semantik
3. *Frame*
4. Aturan(*rules*)

2.6.6 Aturan (*Rules*)

Rules merupakan pengetahuan prosedural yang menghubungkan informasi yang diberikan dengan tindakan (*action*). Struktur *rule*, secara logika menghubungkan satu atau lebih *antecendent (premis)* yang berada pada bagian **IF**, dengan satu atau lebih *consequent (konklusi)* pada bagian **THEN**. Contoh : **IF** warna baju itu biru **THEN** saya suka baju itu.

Sebuah rule dapat mempunyai multiple premise yang tergabung dengan menggunakan operasi logika (**AND, OR**). Bagian konklusi dapat berupa kalimat tunggal atau gabungan dengan menggunakan operasi logika (**AND**) dan dapat pula memiliki kalimat **ELSE**. Contoh : **IF** baju warna biru **AND** harga baju murah, **THEN** saya akan beli baju itu.

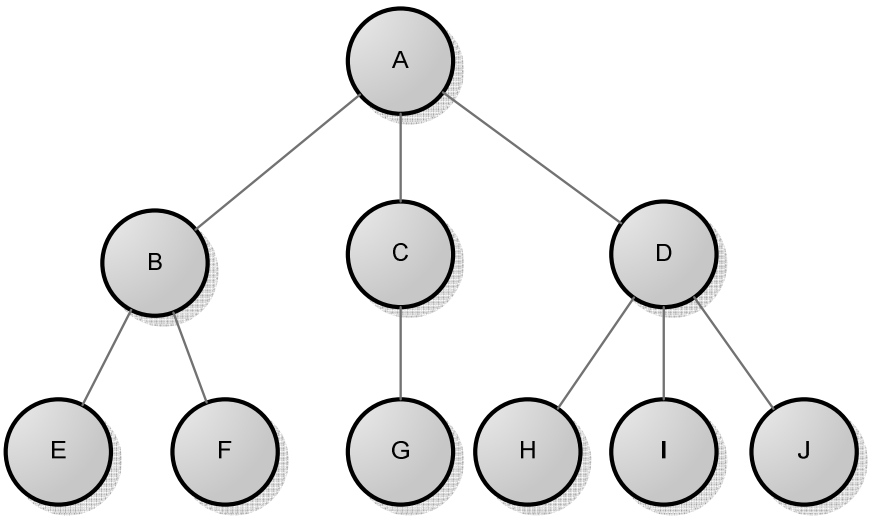
Aturan (*rules*) terdiri dari beberapa jenis, yaitu : relationship/hubungan, rekomendasi, strategi, heuristik, interpretasi, analisis, dan desain. Representasi pengetahuan jenis aturan (*rules*) memiliki beberapa keunggulan dan keterbatasan, yaitu :

1. Keunggulan
    a. Modifikasi dan perawatan relatif mudah.
    b. Uncertainty dapat dikombinasikan dengan *rules.*
    c. Tiap rules biasanya independent dari yang lainnya.
2. Keterbatasan
    a. Pengetahuan yang kompleks membutuhkan rules yang sangat banyak.
    b. Sistem dengan banyak *rules* mempunyai keterbatasan dalam proses pencarian pada bagian program kontrol.

2.7 Metode Inferensi Backward Chaining

Backward chaining adalah suatu rantai yang dilintasi dari suatu hipotesa kembali ke fakta yang mendukung hipotesa tersebut. Pokok permasalahan *backward chaining* adalah untuk mendapatkan suatu rantai yang menghubungkan fakta-fakta ke hipotesis.

Dalam backward chaining, sistem bekerja dari tujuan/hipotesis kemudian menghubungkan fakta yang diperoleh dengan aturan yang ada untuk mencocokkan dengan tujuan yang diharapkan. Gambar 2.9 memperlihatkan konsep dari *backward chaining*.



Gambar 2.9 Konsep Backward Chaining

2.8 Telepon Selular

Handphone atau dengan nama lain telepon selular merupakan sebuah perangkat telekomunikasi elektronik yang mempunyai kemampuan dasar yang sama dengan telepon *fixed line* yang konvensional, namun dapat dibawa ke mana-mana (*portable*) dan tidak perlu disambungkan dengan jaringan telepon

menggunakan kabel (nirkabel, wireless). Indonesia mempunyai dua jaringan telepon selular nirkabel saat ini, yaitu *Global System for Mobile Telecommunication* (GSM) dan *Code Division Multi Access* (CDMA).

Telepon selular selain berfungsi untuk melakukan dan menerima panggilan, umumnya juga mempunya fungsi pengiriman dan penerimaan pesan singkat atau *Short Message Service* (SMS). Telepon selular juga beredar di pasaran dengan harga yang cukup tinggi karena adanya tambahan fitur, seperti kamera, layanan *internet*, *Bluetooth* dan lain sebagainya. Selain itu juga terdapat *manufacture* telepon selular di beberapa negara yang menyiadakan layanan generasi ketiga (3G) yang menambahkan jasa *video call* maupun televisi *online* di ponsel mereka.

Selain kelebihan tersebut, kebanyakan Vendor telepon selular telah menyertakan fasilitas *Java Enable* pada produknya. Dengan adanya fitur *Java Enable* pada telepon selular, memungkinkan pengguna telepon selular tersebut semakin mudah untuk menambahkan sendiri aplikasi-aplikasi berbasisi Java 2 Micro Edition (J2ME).

2.9 Sepeda Motor

2.9.1 Prinsip Kerja Mesin Sepeda Motor

Sistem kerja yang baik pada sebuah mesin kendaraan sangat di tentukan oleh beberapa faktor. Berikut ini adalah beberapa hal yang harus dilakukan agar mesin kendaraan sepeda motor dapat bekerja dengan baik :

1) Mengisap bahan bakar (campuran bensin dengan udara) masuk kedalam ruang bakar.

2) Menaikkan tekanan gas campuran bensin dan udara agar diperoleh tekanan hasil
3) pembakaran yang cukup tinggi.
4) Meneruskan gaya tekan hasil pembakaran sahingga dapat digunakan sebagai tenaga penggerak.
5) Membuang gas hasil pembakaran keluar dari ruang pembakaran.

Pada sepeda motor tenaga didapat dari hasil pembakaran bensin bercampur udara di dalam suatu ruang baker yang kemudian akan menimbulkan panas. Panas ini kemudian di ubah menjadi tenaga gerak/tenaga mekanis di dalam suatu mesin yang disebut "Motor Bakar".

2.9.2 Bagian Penting Mesin Sepeda Motor

1) Silinder

Adalah sebagai tempat pembakaran campuran bahan bakar dengan udara untuk mendapatkan tekanan dan temperature yang tinggi. Silinder pada umumnya dibuat dari bahan baja tuang untuk mesin besar dan untuk mesin kecil terbuat dari bahan logam alumuniun paduan yang di bagian dalamnya dipasangkan tabung dari bahan baja yaitu pada bagian tempat bergeraknya torak.

2) Kepala Silinder

Terbuat dari bahan alumunium paduan, Untuk menghindari terjadinya kebocoran gas terutama pada saat langkah kompresi, maka pemasangan paking dan pengencangan baut untuk merapatkan kepala silinder terhadap silindernya haruslah seteliti mungkin.

3) Bak Mesin

Bak mesin merupakan tempat silinder, poros engkol, dan gigi transmisi, terbuat dari bahan alumunium paduan.

4) Torak/Piston

Terbuat dari bahan alumunium paduan yang mempunyai sifat :

a) Ringan
b) Penghantar panas yang baik
c) Pemuaian kecil
d) Tahan terhadap keausan akibat gesekan dan
e) Kekuatan yang tinggi terutama pada temperature tinggi

5) Cincin Torak

Fungsinya untuk mempertahankan kerapatan antara torak dangan dinding silinder agar tidak ada kebocoran gas dari ruang bakar kedalam bak mesin. Cincin torak ada dua macam, yakni cincin kompresi dan cincin pelumasan.

6) Pena Torak

Berfungsi untuk mengikat torak terhadap batang penggerak, selain itu juga berfungsi sebagai pemindah tenaga dari torak ke batang penggerak agar gerak bolak balik dari torak agar dapat di ubah menjadi gerak berputar pada poros enggkol.

7) Batang Penggerak

Adalah suatu batang yang menghubungkan torak dengan poros engkol

8) Poros Engkol

Pada umumnya poros engkol terbuat dari bahan baja

9) Roda Gaya/Roda Penerus

Berputarnya poros engkol secara terus menerus itu adalah akibat adanya tenaga gerak (energi kinetis) yang di simpan pada roda penerus sebagai kelebihan pada saat langkah kerja. Roda penerus atau disebut juga roda gila dalam pembuatannya harus di balancing dengan teliti agar putaran mesin rata betul tanpa getaran – getaran. Pada mesin sepeda motor umumnya roda gila berfungsi juga sebagai rotor generator.

10) Katup.

Untuk mesin 4 langkah pemasukan bahan bakar dan pembuangan gas sisa pembakaran dilakukan melalui katup masuk dan katup buang.

2.9.3 Prinsip Kerja Mesin 4 Langkah/Tak

1) Langkah Isap

a) Torak dari TMA ke TMB

b) Katup masuk terbuka, katup buang tertutup

c) Campuran bahan bakar dengan udara yang telah tercampur di dalam karburator, masuk dan di isap ke dalam silinder.

d) Saat torak berada di TMB katup masuk akan tertutup

2) Langkah kompresi atau penekanan

a) Torak bergerak dari TMB ke TMA

b) Katup masuk dan katup buang kedua-duanya tertutup sehingga gas yang telah di isap tadi tidak dapat keluar pada waktu di tekan oleh

torak yang mengakibatkan tekanan gas akan naik sambil mengeluarkan panas.

c) Beberapa saat sebelum torak mencapai TMA busi mengeluarkan bunga api listrik.

d) Gas/bahan bakar yang telah mencapai tekana tinggi tadi terbakar

e) Akibat pembakaran bahan bakar tadi, tekanan akan naik menjadi kira-kira tiga kali lipat.

3) Langkah kerja/ekpansi

a) Saat ini kedua katup masih dalam keadaan tertutup

b) Gas yang terbakar tadi dengan temperature dan tekanan yang tinggi akan mengembang kemudian menekan dan memaksa torak turun dari TMA ke TMB

c) Saat inilah pertama kali tenaga poros (kalori) di ubah menjadi tenaga bergerak/mekanis. Tenaga ini disalurkan belalui batang penggerak, selanjutnya oleh poros engkol di ubah menjadi gerak putar

4) Langkah pembuangan

a) Katup buang terbuka, katup masuk tertutup.

b) Torak bergerak dari TMB ke TMA.

c) Gas sisa pembakaran terdorong oleh torak keluar melalui katup buang menuju udara bebas.

2.9.4 Sistem Bahan Bakar

Tenaga yang timbul di dalam suatu motor bakar adalah akibat adanya pembakaran campuran bensin dan udara. Bensin tidak akan mudah terbakar tanpa

adanya oksigen yang tepat di dalam udara. Dengan demikian peranan udara di sini adalah untuk mempermudah pembakaran bensin.

2.9.5 Fungsi Karburator.

Berikut ini adalah fungsi dari karburator :

a) Mencampur perbandingan udara dengan bensin dalam perbandingan yang tepat pada setiap putaran mesin.
b) Memasukkan campuran bensin dengan udara ke dalam ruang bakar dalam bentuk kabut.

2.9.5.1 Fungsi Dari Bagian Karburator

1) Katup gas

Mengatur jumlah campuran bensin dengan udara yang akan di masukkan ke dalam ruang bakar. Bentuk lekukan (coakan) pada bagian bawah katup gas yang mengarah lubang pemasukan udara saat posisi menutup akan berfungsi sebagai penutup saluran udara.

2) Jarum penyiram

Untuk mengatur jumlah campuran bensin dengan udara yang mengalir melalui saluran penyiram dari ¼ sampai dengan ¾ pembukaan katup.

3) Penyiram stasioner

Untuk mengatur jumlah bensin yang digunakan pada waktu stasioner.

4) Penyiram utama

Untuk mengatur jumlah bensin yang digunakan pada waktu putaran tinggi.

5) Ruang pelampung

Sebagai tempat menampung sementara bensin yang akan di alirkan ke ruang bakar.

6) Pelampung

Untuk mempertahankan tinggi permukaan bensin di dalam ruang pelampung agar selalu tetap dan tepat.

7) Sekrup penyetel udara

Untuk mengatur jumlah udara yang akan bercampur dengan bensin.

8) Sekrup penyetel gas

Untuk mengatur posisi pembukaan katup pada dudukan terendah untuk menentukan putaran stasioner.

9) Cuk

Untuk menutup saluran udara agar terjadi percampuran yang kaya dengan sementara apabila menghidupkan mesin pada waktu mesin dan cuaca dalam keadaan dingin.

10) Penyiram udara

Untuk mengontrol udara yang menuju penyiram utama dan kedua agar terjadi percampuran kaya pada kecepatan tinggi dan percampuran miskin pada kecepatan rendah.