BAB 2
TINJAUAN PUSTAKA

2.1. Tinjauan Sekolah

Pada Tinjauan Sekolah ini akan membahas profil dari tempat penelitian yaitu SMK Yapari Aktripa Bandung adalah salah satu Sekolah Menengah Kejuruan (SMK) swasta yang berada di kota bandung, Sekolah mempunyai 3 (tiga) jurusan yaitu Multimedia (MM),Jasa Boga (JB) dan Akomodasi Perhotelan (AP) dan fokus pada bidang pariwisata. Masa Pendidikan di tempuh di SMK Yapari Aktripa dalam waktu tiga tahun, mulai dari kelas 1 (satu) sampai kelas 3 (tiga).

2.1.1. Sejarah SMK

Pendidikan di Indonesia berawal dari Pendidikan berbasis keagamaan yang di selenggarakan oleh para pemuka dan penyebar agama namun sistem Pendidikan dalam bentuk sekolah baru dimulai pada abad ke -16 dan sekolah yang berorientasi "Kejuruan" yang didirikan pertama kali pada zaman VOC adalah akademi Pelayaran (*Academir der Marine*) pada tahun 1743 . Di luar Akademi pelayaran terdapat sekolah pertukangan di Surabaya yang berdiri pada tahun 1853 itulah sebagai sekolah kejuruan pertama di Indonesia. [7]

2.1.2. Sejarah Berdirinya SMK Yapari Aktripa

Menyadari akan tantangan yang di hadapi di era globalisasi pada bentuk persaingan ekonomi yang ketat, maka tenaga-tenaga yang terampil dan siap kerja menjadi sebuah kebutuhan, karena itu, Pendidikan yang berorientasi kerja menjadi alternatif pilihan. Didirikan pada tahun 2008 Yayasan Pariwisata Indonesia – Aktripa (Yapari–Aktripa) melalui SK Pendirian No : 008/SK/YAPARI/XII/2008 pada tanggal 11 Desember 2008, diputuskan untuk dibuka : Sekolah Menengah Kejuruan Yapari Aktripa (SMK Yapari-Aktripa) yang berorientasi pada bidang pariwisata, Sebagai penyelenggara Pendidikan pariwisata tertua di Indonesia memulai SMK Pariwisata

Yapari Aktripa mulai tahun ajaran 2009-2010 dan pada tanggal 9 desember tahun 2010 di berikan ijin operasional dengan Sk No : 421.3/391-DISDIK.[8]

Dengan predikat sekolah yang terakreditasi A, NPSN dan NSS maka SMK Yapari Aktripa menyediakan jembatan emas bagi lulusan SMP/MTs untuk meraih keunggulan dalam pengetahuan, keterampilan, dan kecakapan teknis, sama seperti sekolah vokasi pada umum nya SMK Yapari memiliki beberapa jurusan yang sesuai dengan orientasi sekolah, jurusan tersebut adalah pada bidang perhotelan (AP), Multimedia (MM) dan bidang Jasa Boga (JB) yang siap mengisi pekerjaan di berbagai perusahaan dan instansi serta berwirausaha di bidang-bidang tersebut

Guna menunjang terwujudnya harapan tersebut , dibutuhkan dukungan dari berbagai pihak untuk memudahkan pembelajaran yang dilakukan yaitu dengan memenuhi prasarana yang dibutuhkan sebagai faktor penunjang dengan menyediakan tempat lengkap belajar yang lengkap. Missal nya seperti pada saat praktek kejuruan haruslah ada media atau alat yang dapat digunakan dan instruktur atau guru yang berpengalaman untuk dapat membimbing siswa.

2.1.3. Logo SMK Yapari Aktripa

Logo atau lambang merupakan ciri atau identitas dari sebuah instansi atau organisasi dan setiap logo melambangkan keberadaan, begitu juga dengan SMK Yapari Aktripa yang mempunyai logo tersendiri. Dibawah ini merupakan logo dari SMK Yapari Aktripa.

Gambar 2.1 Logo SMK Yapari Aktripa Kota Bandung

(Sumber : http://www.smkyapariaktripa.sch.id)

2.1.4. Visi dan Misi

Setiap instansi atau organisasi pasti memiliki visi dan misi, sebagai tujuan yang ingin di capai oleh instansi teresebut. SMK Yapari Aktripa pun mempunyai visi dan misi, berikut adalah visi dan misi yang dimiliki oleh SMK Yapari Aktripa

A. Visi

Mewujudkan SMK YAPARI AKTRIPA menjadi, unggul dalam mutu pendidikan dan pelatihan, beriman dan bertaqwa serta mampu bersaing di tingkat nasional dan internasional.

B. Misi

1. Meningkatkan mutu pendidikan dan pelatihan yang memenuhi standar kompetensi nasional dan internasional.
2. Menghasilkan tamatan yang kreatif dan inovatif dalam bidang akomodasi perhotelan, restoran dan Multimedia
3. Menyelenggarakan pendidikan dan pelatihan yang berdasarkan nilai budaya agama dengan mengikuti perkembangan IPTEK

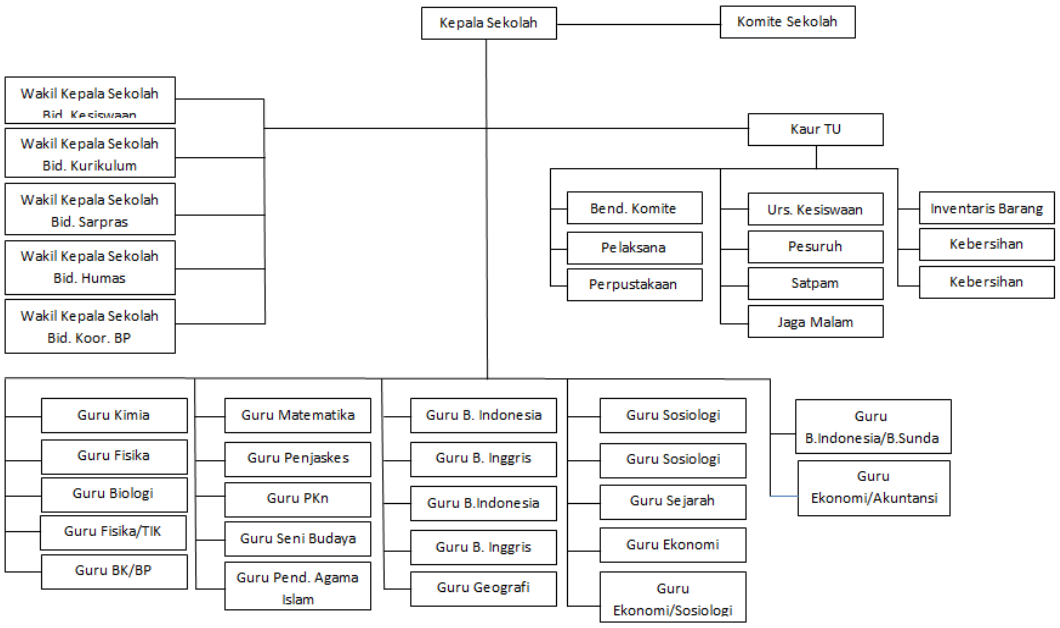
4. Meningkatkan hubungan kerjasama sekolah dengan DU/DI dan instansi lain yang memiliki standar nasional dan internasional.
5. Menerapkan manajemen mutu ISO 9001 : 2000.
6. Melengkapi sarana dan prasarana secara optimal.

2.1.5. Struktur Organisasi

Suatu organisasi pastilah memiliki struktur organisasi nya sendiri, baik itu pada organisasi maupun instansi. Struktur organisasi pada sekolah berperan penting dimana dalam struktur yang ada dapat menjelaskan setiap tugas, peran dan fungsi dari setiap komponen penyelenggaraan organisasi, pada struktur organisasi sekolah merupakan suatu bentuk yang berupa urutan atau daftar yang berfungsi sebagai suatu upaya dalam menjelaskan tugas dan fungsi dari setiap penyelenggara Pendidikan yang bersangkutan dengan sekolah.

Selain sebagai penjelasan mengenai tugas dan fungsi dari setiap komponen yang bersangkutan, struktur organisasi merupaka suatu bentuk susunan yang membedakan antara satu kedudukan dengan kedudukan atau jabatan yang lain, ini di maksudnkan agar adanya rasa hormat antara satu dengan yang lain baik itu berkedudukan di atas, bawah atau setingkat. Jika struktur organisasi dalam sebuah sekolah tidak berjalan dengan baik maka akan adanya ketidak jelasan antara satu tanggung jawab dengan yang lain nya ini menyebabkan aktivitas di sekolah menjadi terhambat atau bahkan dapat menghancurkan organisasi tersebut.

Maka dari itu struktur organisasi ada untuk memudahkan organisasi dalam menjalankan aktifitas sesuai dengan ketentuan yang ada agar dapat melakukan kegiatan sesuai dengan fungsi nya masing-masing dan tidak menyimpang dari yang sudah di tentukan. Berikut ini adalah struktur organisasi di SMK Yapari Aktripa.



Gambar 2.2 Struktur Organisasi SMK Yapari Aktripa

2.2. Landasan Teori

Landasan teori dimaksudkan untuk menjadi pemandu pada saat penelitian agar penelitian yang di lakukan menjadi sesuai dnegan fakta yang ada di lapangan. Selain karena itu landasan teori juga di gunakan sebagai cara untuk memberikan gambaran umum tentang latar penelitian dan juga sebagai bahan pembahasan hasil peneltian. Di mana peneliti mengambil teori teori yang di butuhkan untuk di gunakan pada saat melakukan penelitian.

2.2.1. Kecerdasan Buatan (Artificial Intelligence)

Kecerdasan buatan adalah upaya baru yang menari untuk membuat komputer berpikir, mesin dengan pikiran, dalam arti penuh dan harfiah (Haugeland, 1985), atau pada definisi lain nya dikatakan kecerdasan buatan (*Artificial Intelligence*) adalah salah satu cabang ilmu pengetahuan yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang lebih rumit dengan cara yang lebih manusiawi [9]. Pada hal ini kecerdasan buatan atau AI memanfaatkan agen komputasi untuk di berikan

pengetahuan sedangkan agen adalah sesuatu yang bertindak dalam suatu lingkungan, agent bisa di katakana sebagai komponen yang menjadi penampung pengetahuan yang di berikan, secara langsung serta agen memiliki kemampuan untuk bertindak secara cerdas yaitu ketika :

1. Apa yang di lakukan sesuai dengan keadaan dan tujuannya, dengan mempertimbangkan konsekuensi jangka pendek dan Panjang dari tindakannya
2. Keadaan yang fleksibel untuk mengubah lingkungan dan mengubah tujuan
3. Ia belajar dari pengalaman dan membuat pilihan yang tepat mengingat keterbatasan perseptif dan komputasinya

Secara lebih detail agen komputasi adalah agen yang keputusannya dan tindakannya dapat dijelaskan dalam bentuk perhitungan. Artinya keputusan dapat di pecah menjadi operasi primitive yang dapat diimplementasikan dalam perangkat fisik. Perhitungan ini bisa dalam beberapa bentuk.pada manusia, perhitungan ini dilakukan pada *wetware* dan pada komputer dilakukan di hardware.

Pada awal nya komputer diciptakan hanyalah sebatas alat yang berfungsi untuk menghitung saja, pada perkembangan nya komputer tidak hanya difungsikan untuk alat hitung saja banyak pekerjaan yang dapat diselesaikan menggunakan komputer, oleh karena itu peran komputer semakin diharapkan untuk dapat di berdayakan mengerjakan segala sesuatu yang bisa di kerjakan oleh manusia.Agar komputer bisa bertindak seperti dan menyerupai manusia, komputer perlu di berikan bekal pengetahuan agar mempunyai kemampuan untuk berpikir dan bertindak. Ada beberapa pendefinisian pengertian AI oleh para ahli, dilihat dari bagaimana cara mesin berpikir dengan menggunakan metode berbeda kecerdasan yang berpusat pada manusia harus menjadi bagian dari pengetahuan yang harus di tanamkan pada mesin cerdas.

Berikut adalah beberapa definisi mengenai AI, yaitu:

1) Dilihat dari sudut pandang bagaimana AI berfikir manusiawi, AI adalah upaya baru yang menarik untuk membuat komputer berpikir atau mesin dengan pikiran, dalam arti penuh dan harfiah.
2) Dilihat dari sudut pandang berpikir rational, kecerdasan buatan adalah studi tentang kemampuan mental melalui penggunaan model komputasi serta studi tentang komputasi yang dilakukan memungkinkan untuk merasakan, berakal dan bertindak.
3) Dilihat dari sudut pandang bertindak manusiawi, AI merupakan seni menciptakan mesin yang melakukan fungsi yang membutuhkan kecerdasan sama seperti saat dilakukan oleh orang
4) Dilihat dari sudut pandang bagaimana AI bertindak rasional,Kecerdasan komputasi merupakan belajar bagaimana mendesain agen yang cerdas

2.2.2. Konsep Umum Kecerdasan Buatan

Pengetahuan dari suatu sistem kecerdasan buatan mungkin dapat direpresentasikan dalam sejumlah cara. Salah satu metode yang paling umum untuk merepresentasikan pengetahuan adalah dalam bentuk (rule) **IF…THEN (**Jika..Maka), walaupun cara diatas sangat sederhana namun banyak hal yang berarti dalam membangun sistem kecerdasan buatan dengan mengekpresikan pengetahuan sistem kecerdasan dalam bentuk aturan yang sama.

Ada beberapa konsep yang harus dipahami dalam kecerdasan buatan, diantaranya (kusrini,2006) :

1. Turing Test – Metode Pengujian Kecerdasan

Turing test merupakan sebuah metode pengujian kecerdasan yang dibuat oleh Alan Turing. Proses uji ini melibatkan seorang penanya (manusia) dan dua obyek yang ditanya. Yang satu adalah seorang manusia dan yang satunya

adalah sebuah mesin yang akan diuji. Penanya tidak dapat melihat langsung kepada obyek yang ditanyai sedangkan penanya diminta untuk membedakan mana jawaban komputer dan mana jawabanmanusia berdasarkan jawaban kedua obyek tersebut. Jika penanya tidak dapat membedakan mana jawaban mesin dan mana jawaban manusia maka Turing berpendapat bahwa mesin yang diuji tersebut dapat diasumsikan CERDAS.

2. Pemprosesan Simbolik

   Komputer semula didesain untuk memproses bilangan atau angka-angka (pemprosesan *numerik*). Sementara manusia dalam berfikir dan menyelesaikan masalah lebih bersifat simbolik, tidak didasarkan pada sejumlah rumus atau melakukan komputasi matematika. Sifat penting dari AI adalah bahwa AI merupakan bagian dari ilmu komputer yang melakukan proses secara simbolik dan *non-algoritmik* dalam penyelesaian masalah.

3. Heuristic

   Istilah *heuristic* diambil daribahasa Yunani yang berarti menemukan. Heuristic merupakan suatu *strategi* untuk melakukan proses pencarian (*search*) ruang problem secara selektif, yang memandu proses pencarian yang kita lakukan disepanjang jalur yang memiliki kemungkinan sukses paling besar

4. Penarikan Kesimpulan (*Interferencing*)

   AI mencoba membuat mesin memiliki kemampuan berfikir atau mempertimbangkan (*Reasoning*). Kemampuan berfikir (*Reasoning*) termasuk didalamnya proses penarikan kesimpulan (*interferencing*) berdasarkan fakta-fakta dan aturan dengan menggunakan metode heuristic atau pencarian lainnya.

5. Pencocokan Pola (*Pattern Matching*)

   AI bekerja dengan metode pencocokan pola (*Pattern Matching*) yang berusaha untuk menjelaskan objek, kejadian (*event*) atau proses, dalam hubungan logika atau komputasional.

2.2.3. Ruang Lingkup Kecerdasan Buatan

Langkah pertama dalam menyelesaikan setiap masalah adalah dengan mendefinisikan terlebih dahulu ruang lingkup permasalahan tersebut atau domain untuk permasalahan yang akan di selesaikan. Pada Artificial Intellegence (AI) ruang lingkup di petakan lewat masalah yang dapat terselesaikan dari mesin yang dapat berpikir ini mempengaruhi apa saja komponen-komponen yang di masukkan pada ruang lingkup

Berikut ini adalah ruang lingkup dari Artificial Intelligence(AI), yaitu :

1. Sistem Pakar (*Expert System*). Ditujukan untuk membuat penggunaan secara luas *knowledge* yang khusus untuk penyelesaian masalah tingkat manusia yang pakar.
2. Penglihatan Mesin (*Machine Vision*). Bertujuan pada pengenalan pola dalam beberapa jalan yang sama sebagai kegiatan sistem visual/indera manusia.
3. Robotika (*Robotics*). Difokuskan pada produksi alat-alat mekanik yang dapat mengendalikan gerak. Sebagai contoh : sebuah robot sederhana mampu bergerak/berpindah ke depan,belakang, kanan atau kiri
4. Pengolahan kata (*Speech Processing*). Bertujuan pada pengenalan dan sintesa pembicaraan manusia.
5. *Theorem Proving*. Adalah usaha untuk membuktikan secara otomatis masalah-masalah dalam matematika dan logika.
6. *General Problem Solving*. Bertujuan pada pemecahan kelas-kelas dari masalah-masalah yang ditekankan dalam sebuah bahasa formal.
7. *Pattern recognition*. Difokuskan pada pengenalan dan klasifikasi dari pola-pola

8. Permainan Game (*Game Playing*). Pembuatan program-program bermain permainan.

2.2.4. Bahasa Alami (Natural Language)

Bahasa sendiri adalah kapasitas khusus yang ada pada manusia untuk meperoleh dan menggunakan sistem komunikasi yang kompleks dan sebuah bahasa adalah contoh spesifik dari sistem tersebut. Pada dasarnya bahasa terbagi menjadi dua (1) Bahasa Alami, dan (2) bahasa buatan.Bahasa alami adalah bahasa yang dipelajari manusia dari lingkungannya yang digunakan untuk berkomunikasi dengan manusia atau dalam pengertian lain bahasa alami jika dikaitkan dengan kecerdasan buatan adalah bahasa yang dibuat oleh manusia untuk berkomunikasi dengan teknologi komputer dengan menggunakan bahasa manusia.Bahasa buatan merupakan bahasa yang di susun berdasarkan pertimbangan akal untuk untuk menyampaiikan sebuah konsep tertentu. Dalam bahasa buatan, simbol yang berlaku sebagai pengandung arti di sebut "istilah" sedangkan yang dikandung oleh istilah adalah sebuah konsep. Bahasa buatan bisa dibedakan atas dua jenis bahasa, yaitu bahasa istilah dan bahasa artifisial

a. Bahasa istilah, bahasa istilah adalah bahasa yang rumusannya diambil dari bahasa biasa yang diberi arti tertentu, misalnya "sosiologi" berasal "sosio" yang berarti masyarakan dan logos yang berarti ilmu. Jika kedua term ini maknanya digabung, maka akan menjadi sebuah bahasa tertentu yang ditunjukan terhadap arti "lmu tentang masyarakat.
b. Bahasa artifisial, bentuk bahasa ini merupakan pemahaman tentang simbol-simbol, dengan kata lain, bahasa ini termasuk bahasa buatan yang bersifat simbolik. seperti yang sering digunakan oleh para ahli matematika ahli fisika, dan lain sebagainya.

Chomsky adalah orang yang berkontribusi besar dalam merepresentasikan Bahasa sebagai rangkaian symbol untuk pertamakali. Kontribusi besar Chomsky untuk Linguistik secara luas dianggap sebagai teori sintaksis matematika, yang kemudian

diperluas untuk menjelaskan struktur semantik. Chomsky mengusulkan abstrak, teori matematika bahasa yang memperkenalkan model generatif yang menyebutkan (tak terhingga banyak) kalimat dalam bahasa. Model ini terdiri dari aturan produksi Postlike, yang secara formal mengubah urutan simbol, dan bentuk tidak terbatasnya setara dengan cara komputasi Turing, meskipun pembatasannya setara dengan automata yang lebih sederhana. Formalisme mengandung simbol non-terminal yang berhubungan dengan frase sintaksis, seperti frase kata kerja, dan terminal yang biasanya kata-kata, huruf, atau blok bangunan lainnya ini yang di namakan pemprosesan Bahasa alami (*Natural Language Processing*).

Linguistik sendiri adalah studi ilmiah tentang Bahasa.berbeda dengan disiplin ilmu lain yang berhubungan dengan Bahasa, linguistic berkepentingan dengan menggambarkan struktur Bahasa yang di atur oleh aturan, dan leksikon atau perbendaharaan kata adalah salah satu pembahasan pada bidang studi ini.

Perbendaharaan kata atau *vocabulary* adalah sekumpulan kata dan frase yang di gunakan pada Bahasa yang dipakai untuk berkomunikasi contoh seperti Bahasa Indonesia,Bahasa inggris dan lainya.kosa kata di maksudkan sebagai bagian dari pengkajian Bahasa, dan pada linguis pendefinisian kata-kata dan frase-frase yang digunakan secara umum adalah dengan mengumpulkannya ke dalam sebuah leksikon. Leksikon sebagai kamus yang mengorganisasikan kata-kata bahasa secara alfabet.

2.2.5. Pengolahan Bahasa Alami (*Naturan Language Processing*)

Natural Language Processing adalah bidang penelitian dan aplikasi yang mengekplorasi bagaimana komputer dapat digunakan untuk memahami dan memanipulasi teks atau pidato Bahasa alami untuk melakukan hal-hal yang bermanfaat [10]. NLP tidak memperdulikan bagaimana sebuah kalimat dimasukkan ke komputer tetapi mencopy informasi dari kalimat tersebut

Dibawah ini adalah cara agar mesin dapat menyerap informasi dari sebuah kalimat, yaitu;

a. Pendekatan-pendekatan pada *NLP*. Inti dari *NLP* adalah PARSER, dimana PARSER tersebut membaca setiap kalimat, kata demi kata, untuk menentukan apa yang dimaksud .PARSER sendiri terdiri dari 3 jenis:.

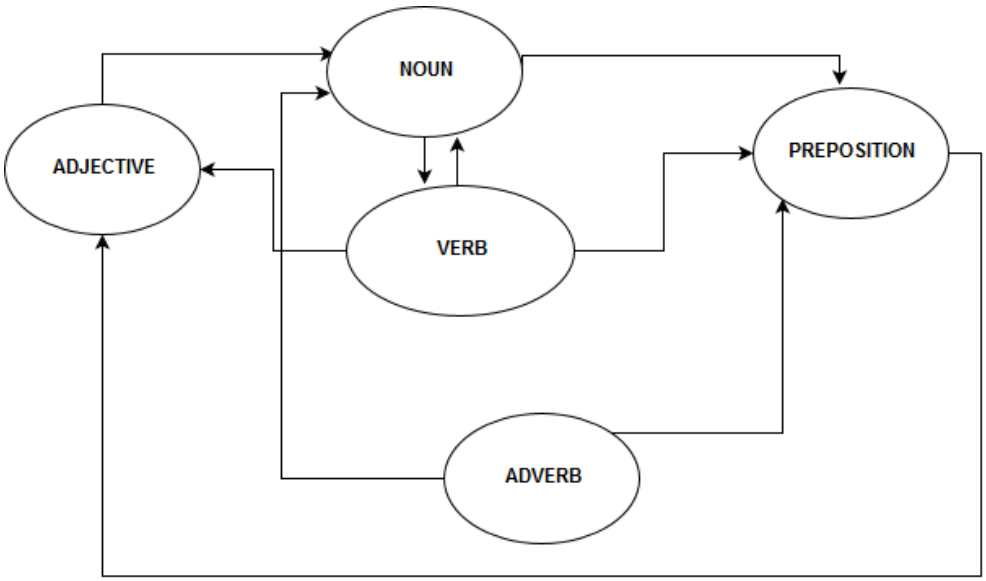
1. PARSER STATE-MACHINE
2. PARSER CONTEXT-FREE RECURSIVE-DESCENT
3. PARSER NOISE-DISPOSAL

b. Batasan Bahasa

Aspek yang paling sulit dalam pembentukan sistem pengendali NLP adalah pengakomodasian kekompleksan dan kefeksibelan Bahasa manusia dalam sistem [13].

2.2.5.1. Parser State Machine NLP

Parser State-Machine menggunakan keadaan yang sesungguhnya dari kalimat untuk memprediksi tipe apa yang berlaku.*State-Machine.Directed graph* yang menunjukan bagaimana transisi yang valid dari satu state ke state yang lain.contoh grammar G1.



Gambar 2.3 State Machine Grammar G1

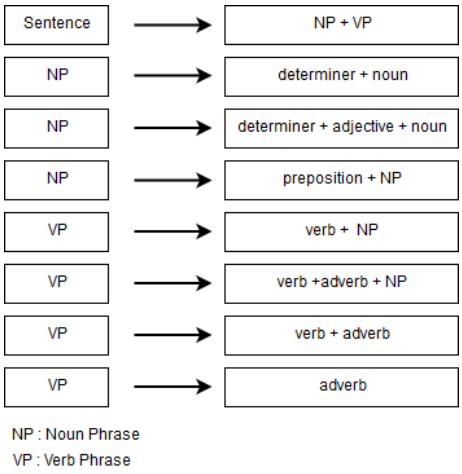
Kegunaan *state-machine* adalah dapat memotong atau memilah kalimat ke dalam komponen-komponennya serta dapat menentukan apakah sebuah kalimat dibentuk dengan benar dalam Batasan dari *Grammar* G1, setelah memotong grammar kemudian berlanjut ke proses bagaimana menentukan database yang harus dibentuk dengan benar sebelum implementasi yaitu dengan membentuk kosa kata (*vocabulary*) dari kata-kata yang dikenal kesistem dengan mengikuti tipe yang ada kemudian menyimpan keadaan sesungguhnya dari kalimat

Masalah yang paling buruk dengan *state-machine parser* adalah kekompleksannya yang dalam grammar G1 di butuhkan 14 clause yang terpisah untuk menunjukkan transisi keadaan serta hal yang paling sulit adalah parser tidak mengetahui bagaimana mencapai suatu keadaan seperti contoh : parser tidak dapat menghubungkan sebuah modifikasi phrase ke noun tertentu. Hal ini berarti tidak dapat memanggil *parser state-machine* untuk mendukung suatu informasi lain dari keadaan yang sesungguhnya

2.2.5.2. Parser Context Free Recursive Descent

Pendekatan ini menggunajan menggabung kan item-item sanmpe di potong ke pada elemen-elemenya contoh nya : sebuah kalimat adalah gabungan dari berbagai item dan item ini adalah gabungan dari item lain dan seterusnya sampai dipotong (dipilah) ke elemen-elemennya seperti Noun, Adjective, dan sebagai nya.Aturan-aturan yang ada pada setiap bagian yang telah dibentuk disebut *Production rule* dari *grammar* sedangkan parser *context free* menggunakan production rule untuk menganalisa sebuah kalimat.

Production Rule untuk grammar G1 seperti gambar di bawah ini :



Gambar 2.4 Production rule untuk grammar

Tata Bahasa atau grammar dapat digunakan untuk mengubah kalimat sehingga hanya kalimat yang benar secara gramatikal yang dihasilkan mulai dari simbol khusus; yaitu, struktur frasa dari sebuah kalimat dengan mudah dinotasikan dengan bantuan tata bahasa seperti itu, suatu proses yang dikenal sebagai derivasi

Berlanjut ke sintaks kalimat struktur yang sama dapat digunakan untuk mengenali kalimat dalam bahasa, prosedur yang disebut sebagai parsing. Proses analisis dan pembuatan sintaks sangat penting untuk pemrosesan bahasa, karena memungkinkan kita untuk melanjutkan ke tahap kedua dalam menafsirkan bentuk sintaksis dan membangun representasi semantik, yang merupakan tugas memahami teks, tugas ganda yang menghasilkan sebuah ekspresi dari representasi semantik. Keseluruhan keseluruhan pengolahan tersebut dikenal sebagai Pemahaman Bahasa Alami.

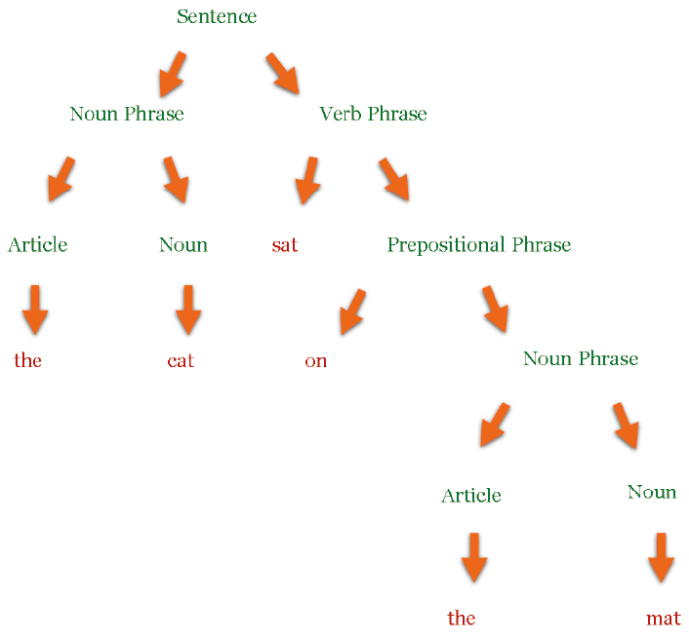
$S \rightarrow NP\ VP$ $\quad$ $A \rightarrow a \mid the$

$NP \rightarrow A\ N$ $\quad$ $VP \rightarrow V\ VP$

$V \rightarrow sat \mid meowed \mid slept \mid \ldots$ $\quad$ $N \rightarrow table \mid chair \mid cat \mid mat \mid \ldots$

Gambar 2.4 Notasi Grammar

Tata Bahasa yang di notasikan seperti Gambar 2.6, yang menunjukan contoh yang tidak lengkap. Simbol nonterminal adalah singkatan dari Gambar 2.4 dibawah ini menggambarkan derivasi yang dari kalimatnya lengkap mulai dari symbol non-terminal diawal kalimat , panah menunjukan penerapan aturan produksi, nama lengkap digunakan untuk symbol non-terminal, symbol terminal berwarna merah.



Gambar 2.5 contoh derivasi dari kalimat "*the cat sat on mat*"

Parser membentuk tipe parse *tree* yang disebut *context free* sebab *Tree* bukan dasar dari konteks setiap elemen, hal ini berarti bahwa aturan atau *rule* akan bekerja untuk suatu *statement* yang menyerupai *Grammar* G1 tanpa menpa mengharapkan pada konteks setiap phrase.

2.2.5.3. Parser Noise Disposal

Tipe parser ini sesungguhnya sangat umum dalam aplikasi tipe database, seperti *command processor*. *Command processor* adalah program yang di panggil monitor terminal (TMP) ketika pengguna di terminal memasukkan nama perintah sedangkan TMP adalah program yang menerima dan menafsirkan perintah, dan yang membuat perintah prosesor sesuai dengan yang dijadwalkan dan dijalankan selain itu TMP juga berkomunikasi dengan pengguna terminal, menanggapi penghentian abnormal dan memproses interupsi.

Contoh seperti sebuah database yang terdiri dari nama-nama perusahaan dan harga-harga stock dan asumsikan database menerima query seperti ini :

- Lihatkan saya semua perusahaan dengan persediaan > 100
- Lihatkan saya semua
- Lihatkan saya xyz
- Lihatkan saya satu dengan persediaan < 100

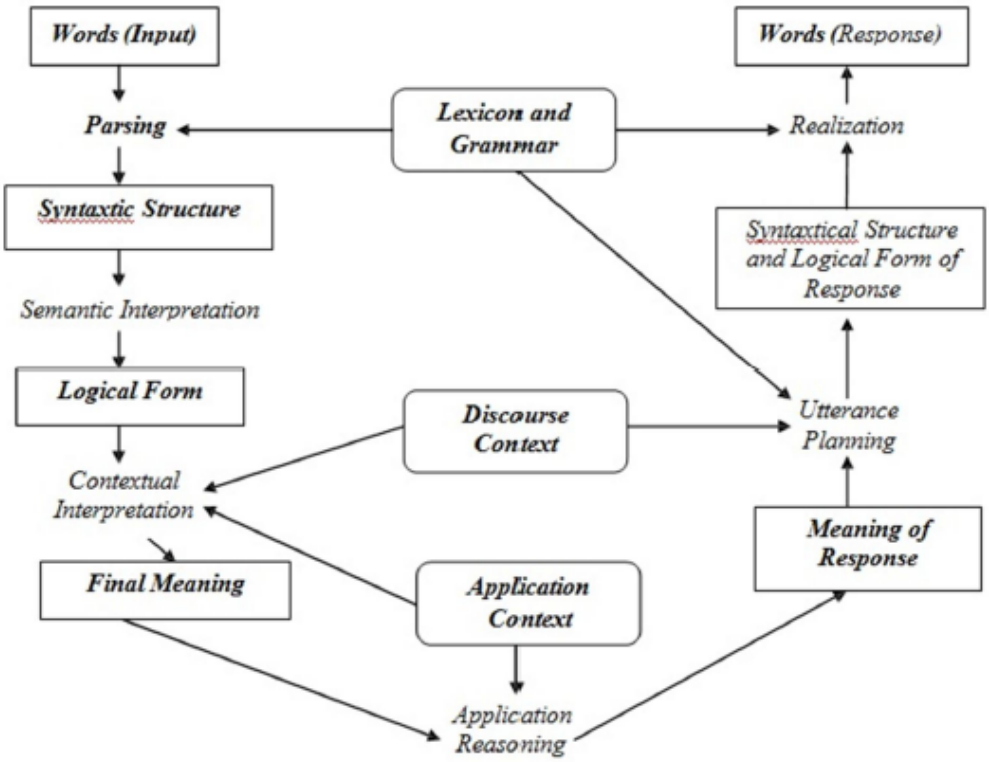
Tipe query yang terjadi adalah : Perintah <modifikasi><nama><operator><nilai>, maka dari itu perintah harus selalu ada, tetapi 4 elemen lainnya adalah optional walaupun begitu bila operator digunakan maka nilai harus tetap digunakan.

2.2.5.4. Tahapan-Tahapan pada NLP

Untuk membuat NLP (*Natural Language Processing*) dapat bekerja memahami apa yang pengguna masukkan maka gramatika dan definisi kata-kata haruslah bisa cocok dengan pengetahuan internal dan pengolahan inpu. Pelacakan klasik dan Teknik pencocokan (pattern matching) digunakan Bersama dengan basis pengetahuan yang ada agar Bahasa alami yang di gunakkan mesin mengerti ucapan yang pengguna berikan, jika mesin sudah dapat meberikan hasil yang pengguna inginkan/respon maka kembali harus kembali di nyatakan atau di ekspresikan dalam Bahasa alami kembali

Untuk itu terdapa beberapa tahapan yang dibutuhkan untuk mencapai tujuan tersebut, proses yang pertama adalah parsing atau analisis sintaksis, sama seperti yang sudah dijelaskan di atas bahwa proses parsing adalah memeriksa kebenaran suatu struktur kalimat berdasarkan dari tata Bahasa (*Grammar*) dan kosa kata (*lexicon)* tertentu[11].

Proses selanjutnya adalah interpretasi semantik (*semantic interpretation*) yang dimaksudkan untuk meintepretasikan arti dari kalimat tertentu secara *context-independent* untuk kebutuhan lebih lanjut dan proses yang terakhir adalah intepretasi kontekstual (*contextual interpretation*) yang berguna untuk merepresentasikan arti secara *context-dependent* serta dapat menentukan maksud dari penggunaan kalimat, pada di bawah ini terdapat contoh gambar dari sebuah organisasi sistem NLP yang dinotasikan pada gambar 2.6 berikut ini:



Gambar 2.6 Organisasi Sistem NLP

Jenis aplikasi-aplikasi dari bidang pengolahan Bahasa alami (NLP), berikut adalah contoh-contoh tugas yang sering sering digunakan dalam kemampuan NLP, seperti :

1. Kategorisasi konten (*content categorization*). Ringkasan dokumen berbasis linguistikm termasuk pencarian dan pengindeksan, pemberitahuan kontendan deteksi duplikasi
2. Penemuan dan pemodelan topik (*Topic discovery and modeling*). Menangkap inti dan tema secara akurat dari koleksi teks, dan menerapkan analitik tingkat lanjut ke teks, seperti pengoptimalan dan memperkirakan.
3. Ekstraksi kontekstual (*contextual extraction*). Secara otomatis menarik informasi terstruktur dari sumber berbasis teks.
4. Analisis sentiment (*Sentiment Analysis*). Mengidentifikasi suasana hati atau opini subjektif dalam jumlah besar pada teks, termasuk sentimen rata-rata dan penarikan opini.
5. Konversi ucapan-ke-teks dan teks-ke-ucapan (*Speech-to-text and text-to-speech conversion*). Mentransmisikan perintah suara menjadi teks tertulis, dan sebaliknya.

2.5.1. Question Answering System

QA (*Question Answear*) dapat didefinisikan sebagai proses otomatis mampu memahami pertanyaan yang dirumuskan dalam bahasa alami seperti bahasa Inggris dan merespons persis dengan informasi yang diminta. secara detail karakteristik dan fungsi sistem QA yang ideal seharusnya. Sistem ini seharusnya bisa untuk menentukan kebutuhan informasi yang diungkapkan dalam sebuah pertanyaan, cari informasi yang diperlukan, ekstrak itu, dan kemudian menghasilkan jawaban dan menyajikannya sesuai dengan persyaratan yang diungkapkan dalam pertanyaan.Selain itu, sistem yang

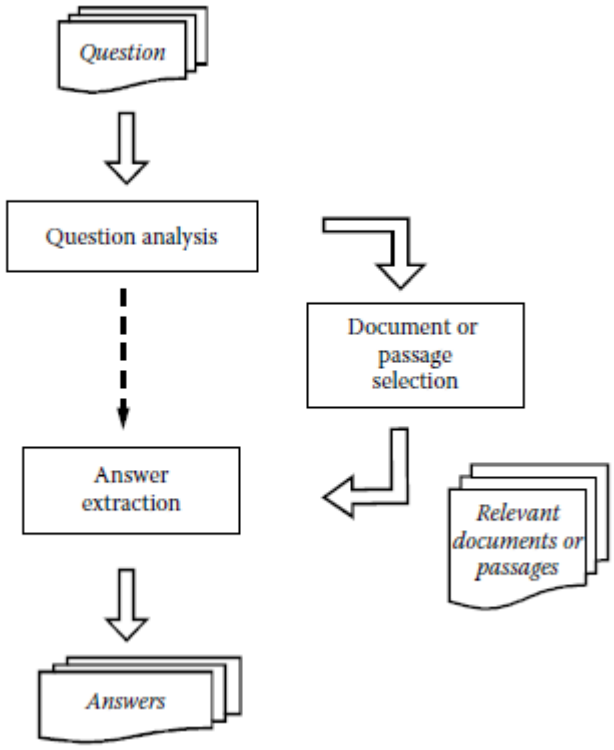
ideal ini harus mampu menafsirkan pertanyaan dan memproses dokumen yang ditulis bahasa alami yang tidak dibatasi domain, yang akan memungkinkan interaksi yang nyaman dan sesuai oleh pengguna[12]

persyaratan sistem QA yang ideal dan berkonsentrasi pada upaya penyelesaian masalah yang lebih khusus dan lebih mudah dikelola seperti memilih dokumen berdasarkan kebutuhan informasi tertentu, Minat dalam domain terbuka QA muncul dari asumsi bahwa pengguna akan lebih memilih jawaban yang tepat pertanyaan pengguna daripada harus memeriksa semua dokumentasi yang tersedia terkait dengan topik pencarian untuk memenuhi kebutuhan informasi itu sendiri.terdapat dua dokumen yang berusaha untuk mengatur penelitian dipada QA yang pertama pernyataan visi untuk memandu penelitian dalam pertanyaan menjawab (tanya jawab) dan ringkasan teks (Carbonell, 2000) dan Masalah, tugas dan struktur program untuk penelitian *roadmap* dalam pernyataan dan menjawab (burger, 2000)

pada dasarnya dokumen pertama mengatur ruang lingkup dan kemampuan sistem QA di masa yang akan mendatang untuk dapat memenuhi harapan dan persyaratan dari segi spektrum yang luas dari pengguna potensial yang berbeda. Dari pengguna biasa sampai kepada analis informasi yang professional dan dokumen kedua mendefinisikan roadmap yang dimaksudkan untuk membahas dan memungkinkan visi untuk penelitian yang ada dalam dokumen pertama. dalam modul analisis pertanyaan-pertanyaan yang diajukan ke sistem proses di gunakan untuk mendeteksi dan mengekstrak informasi yang berguna untuk modul-modul lain. Ini dilakukan ileh dua tugas utama yaitu :

1. Klasifikasi pertanyaan untuk menentukan jenis informasi yang diharapkan oleh pertanyaan sebagai jawaban
2. Pemilihan elemen-elemen yang akan memungkinkan sistem untuk mencari dokumen yang kemungkinan berisi jawabannya

dari proses awal ini sangat penting sejak kinerja yang di lakukan modul yang tersisa dengan ekstensi, dan keseluruhan sistem akan bergantung pada kualitas dari informasi yang diambil dari pertanyaan.bahkan derivasi yang salah dari jawaban yang diharapkan, disebutkan bahwa 36,4% dari kesalahan dalam sistem yang pernah digunakan adalah karena derivasi yang salah dari jenis jawaban yang di harapkan (Moldovan, 2003).



Gambar 2.7 Sistem arsitektur umum QA

2.5.2. Scanner (Analisis Leksikal)

Analisis Leksikal merupakan antarmuka antara kode program sumber dan analisis sintaktik (parser). Scanner melakukan pemeriksaan karakter per karakter pada teks masukan, memecah sumber program menjadi bagian-bagian disebut Token.pada satu sisi Analisis ini melakukan penerjemahan masukan menjadi bentuk yang lebih

berguna untuk tahap-tahap kompilasi berikutnya. Analisis Leksikal mengerjakan pengelompokkan urutan-urutan karakter ke dalam komponen pokok: identifier, delimeter, simbol-simbol operator, angka, keyword, noise word, blank, komentar, dan seterusnya menghasilkan suatu Token Leksikal yang akan digunakan pada Analisis Sintaktik. Terdapat 2 aspek penting pembuatan analisis leksikal, yaitu :

a. Menentukan token-token Bahasa
b. Mengenali token-token Bahasa dari program sumber

Token-token dihasilkan dengan cara memisahkan program sumber tersebut dilewatkan ke parser, selain itu juga Analisis Leksikal harus mengirim token ke parser. Untuk mengirim token, scanner harus mengisolasi barisan karakter pada teks sumber yang merupakan 1 token valid. Scanner juga menyingkirkan informasi seperti komentar, blank, batas-batas baris dan lain-lain yang tidak penting (tidak mempunyai arti) bagi parsing dan Code Generator.Analisis leksikan juga menjadi komponen kompilasi independent yang berkomunikasi dengan parser lewat antar muka yang terdefinisi bagus dan sederhana sehingga pemeliharaan analisi leksikal menjadi lebih mudah dimana perubahan-perubahan terhadap analisis leksikal tidak berdampak pada pengubahan kompilator secara keseluruhan.

Pada analisis leksikal yang dituntun tabel (table-drivern lexical analyzer), maka satu satunya yang berubah adalah table itu sendiri, Untuk itu perlu komunikasi tingkat lebih tinggi yang biasanya dilakukan suatu struktur data dipakai bersama seperti tabel simbol.pada tugas yang harus di kerjakan oleh analisis leksikal adalah meliputi Konversi Program Sumber Menjadi Barisan Token Mengubah program sumber yang dipandang sebagai barisan byte/karakter menjadi token dan juga Menangani Kerumitan Sistem Masukkan/Keluaran

Karena analisis leksikal biasanya berhubungan langsung dengan kode sumber yang diwadahi file, maka analisis leksikal juga bertindak sebagai benteng untuk komponen-komponen lain di kompilator dalam mengatasi keanehan-keanehan sistem

masukkan/keluaran sistem operasi dan sistem komputer. Kemudian hasil yang di dapat dari penganalisa akan digunakan oleh penganalisa sintaks yang akan memeriksa urutan dari simbol-simbol kelas kata atau *lexicon* didalam kalimat.

Tahapan yang ada pada analisis leksikal meliputi beberapa proses berikut proses yang harus di lakukan oleh analisis leksikal :

1. Pengenalan Token

   Scanner harus dapat mengenali token yang kemudian di lanjutkan untuk mendeskripsikan token-token yang harus dikenali
2. Pendeskripsian Token

   Menspesifikasikan aturan-aturan pembangkit token-token dengan kelemahan reguler grammar menspesifikasikan token berbentuk pembangkit, sedang scanner perlu bentuk pengenalanmenggunakan ekspresi grammar dan menspesifikasikan token-token dengan ekspresi regular .
3. Implementasi Analisis Leksikal

   Pada pemodelan analisis leksikal sebagai pengenal yang menerapkan finite automata, analisis leksikal tidak cuma hanya melakukan mengatakan YA atau TIDAK. Dengan demikian selain pengenal, maka analisis leksikal juga melakukan aksi-aksi tambahan yang diasosiasikan dengan string yangsedang diolah
4. Penanganan Kesalahan di Analisis Leksikal

   Hanya sedikit kesalahan yang diidentifikasi di analisis leksikal secara mandiri karena analisis leksikal benar-benar merupakan pandangan sangat lokal terhadap program sumber.

Pada Aplikasi yang akan dibuat pada sitem cerdas NLP, program sumber yang dimaksud adalah program sumber yang diolah oleh *scanner* berupa kalimat input dari pengguna berbentuk karater atau string didalam chat.

Ketika *scanner* meneerima input berupa *stream* karakter yang ada kemudian memilah bagian saruan leksik, kemudian satuan leksik yang ada terdiri dari symbol-simbol satuan yang jika di kombinasikan akan dapat mempunyai arti yang berbeda-beda. Symbol yang dapat dioergunakan dalam suatu Bahasa tertentu terbatas jumlahnya. Symbol-simbol

Sebagai contoh dalam pembicaraan grammar, anggota alfabet dinamakan symbol terminal atau token. Klimat adalah string yang terurut atas symbol-simbol terminal sedangkan Bahasa adalah suatu himpunan kalimat-kalmat.Pada bagian Bahasa bisa berupa tak terhingga hingga kalimat.

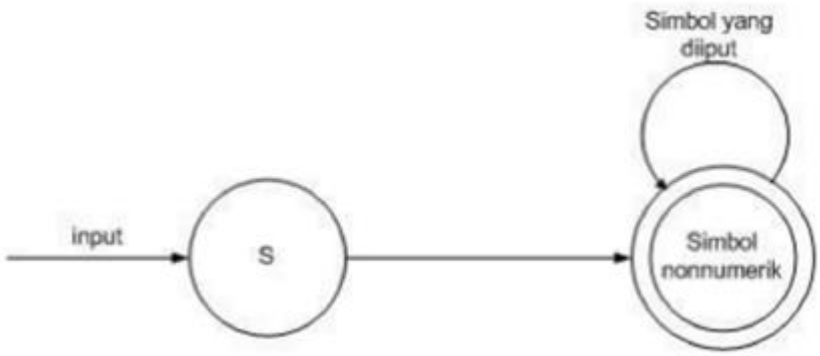
Symbol-simbol berikut adalah symbol terminal :

1. Huruf kecil pada alphabet, misalnya : a,b,c
2. Symbol operator, misalnya : +,-, dan /
3. Symbol tanda baca,misalnya : (,),dan :

Sednagkan symbol berikut merupakan symbol non-terminal:

1. Huruf besar awal alphabet, misalnya : C,D,E
2. Huruf S sebagai simbol awal

Pada gambar 2.8 di bawah ini akan di terlihat pada saat scanner membaca input, tools yang digunakan menggambarkan perpindajan dari posisi lainnya adalah diagram transisi:



Gambar 2.8 Diagram transisi

2.5.3. *Parser* (Analisis Sintaksis)

Bahasa komputer biasanya dirancang untuk memungkinkanencoding oleh tata bahasa yang tidak ambigu dan parsing dalam waktu linier dari panjang input.

Pada *Question Answering system*, fungsi dari parser tidak sama dengan yang ada pada kasus lain dikarenakan seriap *token* yang diolah semua memiliki tipe data yang sama satu sama lain yaitu berupa kata (word). Adapun urutan kemunculan dari sebuah *token* yang berupa beberapa kata itu akan di olah dengan berdasarkan dari *brainfile. Sedangkan kemmpuan* parser untuk mengolah *token* dan bekerja sama dengan *brainfile* ini yang paling menentukan tingkatan kexerdasan dari sebuah *chat bot*

2.5.4. Parsing (proses Penurunan)

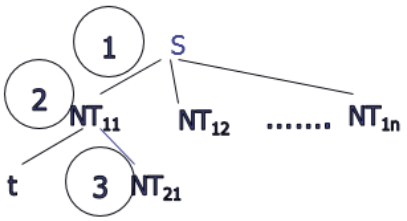
Parsing adalah kontruksi atau pembentukan pohon sintaks untuk suatu kalimat (ekspresi).bila terdapat lebih dari satu pohon sintak untukj sebuah *grammar* maka dikatakan *grammar* tersebut *ambiguous* (ambigu),terdapat du acara untuk melakukan validitas sintaks dengan parsing, yaitu

1. *Top Down Parsing*

   Dengan cara top down parsing, yai tu melakukan derivasi pada string string dengat NT.contoh nya adalah jika a adalah input string, maka derivasi dari top down parse dapat di tunjukkan sebagai brikut :

   S ⟶ … ⟶ … ⟶ … ⟶ a

   Parse tree untuk top down parsing selalu dimulai dari sebelah kiri, seperti contoh yang ada pada gambar di bawah ini :

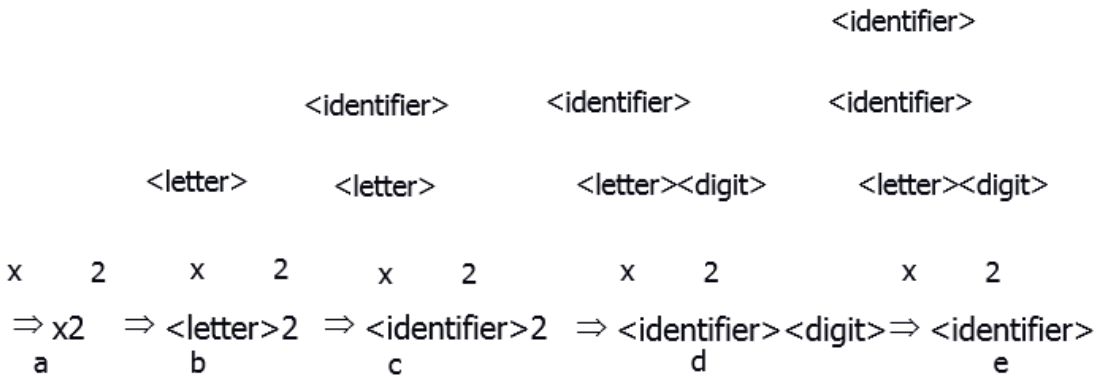


Gambar 2.9 Parsing Top Down

2. Parsing Bottom Up

   Parsing Bottom Up membangun pohon sintaks melalui urutan simbol yang direduksi, atau dimulai dengan sebuah string hingga mencapai simbol start Grammar.

   Contoh : jika diketahui identifier x2, dengan parsing bottom up akan menjadi :

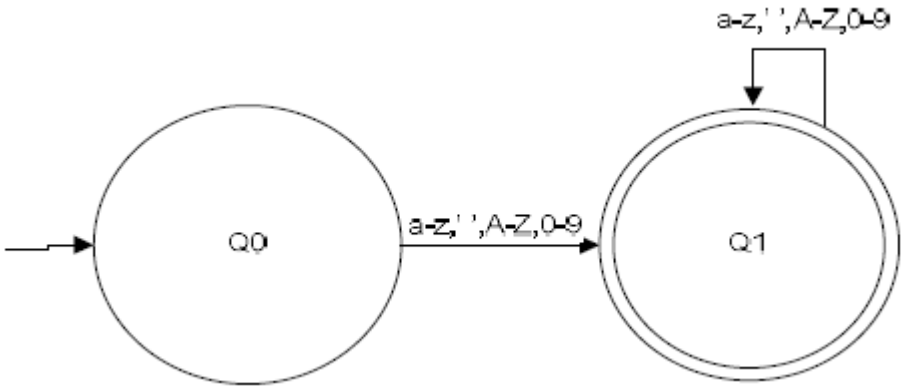


Gambar 2.10 Parsing Bottom Up

Proses *parsing* atau penguraian kalimat juga merupakan proses yang berjalan untuk menterjemahkan inputan dari user agar dapat dikenali oleh sistem, secara *default*, semua kalimat masukkan pengguna yang di inputkan akan di anggap sebagai kata-kata yang harus ada pada data yang akan dicari. Missal ada pengguna memasukkan kalimat berikut:

Penggunaan Bahasa alami

Maka sistem akan melakukan pencarian pada data yang ada di dokumen yang memiliki kata Bahasa atau alami, pada dasarnya jika terdapat dokumen yang mempunyai kata Bahasa inggris maka dokumen tersebut dianggap sesuai dengan yang dicari oleh pengguna, karena mengandung kata Bahasa. Pada hal ini menjadikan pengguna yang akan mencari informasi mengenai Bahasa alami tidak akan mendapat hasil yangs sesuai dengan yang diinginkandan jika pengguna melakukkan proses pencarian lebih spesifik, agar indformasi yang di dapat lebih cocok pada yang di inginkan, maka pengguna harus mengikuti proses aturan proses penguraian kalimat yang di miliki sistem seperti dibawah ini :



Gambar 2.11 Proses Parsing Finite State Auotomata

Konfigurasi FSA di atas secara formal di nyatakan sebagai berikut ini :

Q = {Q0, Q1}

Σ = {a-z,' ',A-Z,0-9}

S = Q0

F = {Q1}

Fungsi transisi yang ada sebagai berikut :

□(Q0, a-z|' '|A-Z|0-9) = Q1

□(Q1, a-z|’ ‘|A-Z|0-9) = Q1

Fungsi di atas dapat disajikan dan dilihat pada table di bawah ini

□	a-z|’ ‘|A-Z|0-9
Q0	Q1
Q2	Q1

Tabel 2. 1 Tabel Parsing

Dari diagram state gambar 2.11 , didapat aturan produksi sebagai berikut:

S ->aA..zA| AA..ZA | 0A-9A|<space>A|a..Z|A..Z|0..9|<space>

A ->aA..zA| AA..ZA |0A-9A|<space>A|a..Z|A..Z|0..9|<space>

Dimana secara formal dapat ditulis :

V = {S, A}

T = {a-z,’ ‘ A-Z,0-9}

22

P = {S->aA..zA|AA..ZA|0A..9A|<space>A|a..Z|A..Z|0..9|<space>,

A->aA..zA| AA..ZA |0A..9A|<space>A|a..Z|A..Z|0..9|<space>}

S = S

Aturan produksi diatas menggunakan left to right sehingga jika pengguna memasukan pertanyaan “hai juga”, maka proses akan dilakukan dari depan (left) ke belakang (right), seperti berikut: :

S ->hA

S ->haA

S ->haiA

S ->haiA

S ->hai jA

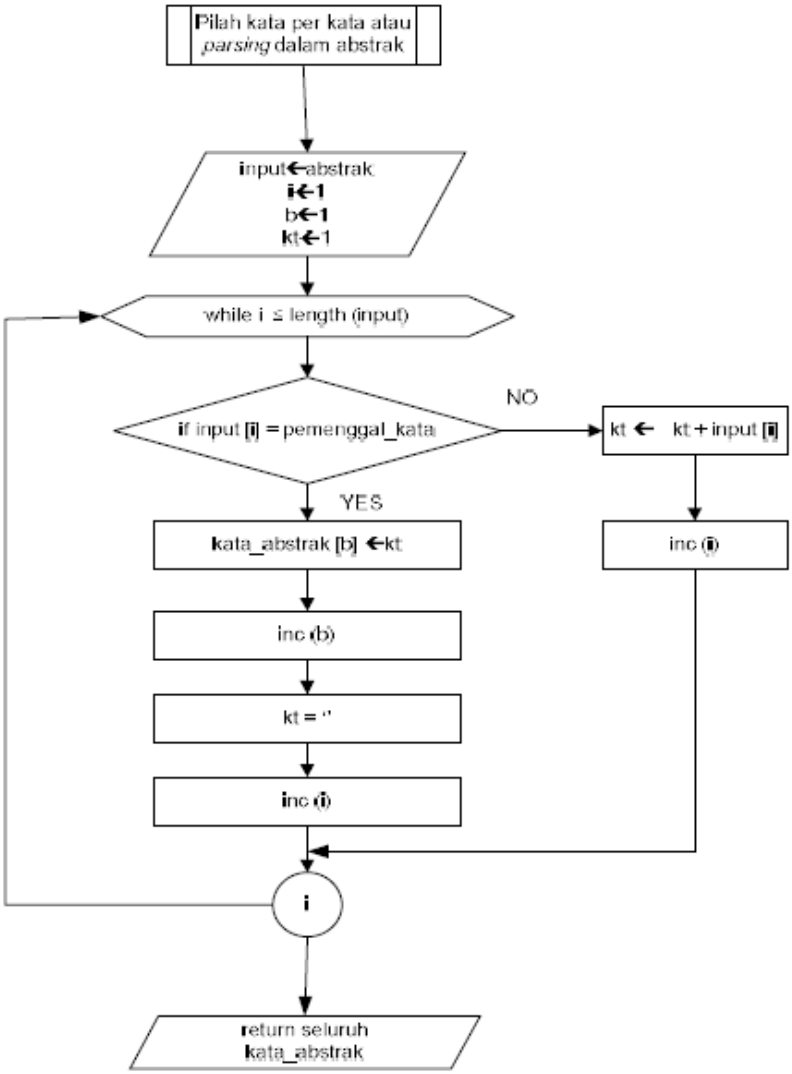
S ->hai juA

S ->hai jugA

S ->hai jugaA

S ->hai juga

Pada proses nya setiap kalimat atau kata di pecah menjadi beberapa bagian karakter dalam bentuk array. Di bawah ini akan digambarkan bagaimana proses parsing abstrek lewat *flowchart* yang dapat dilihat di gambar 2.13, proses ini dipakai untuk memotong – motong dokumen per kata dan akan disimpa pada array kata abstrak.

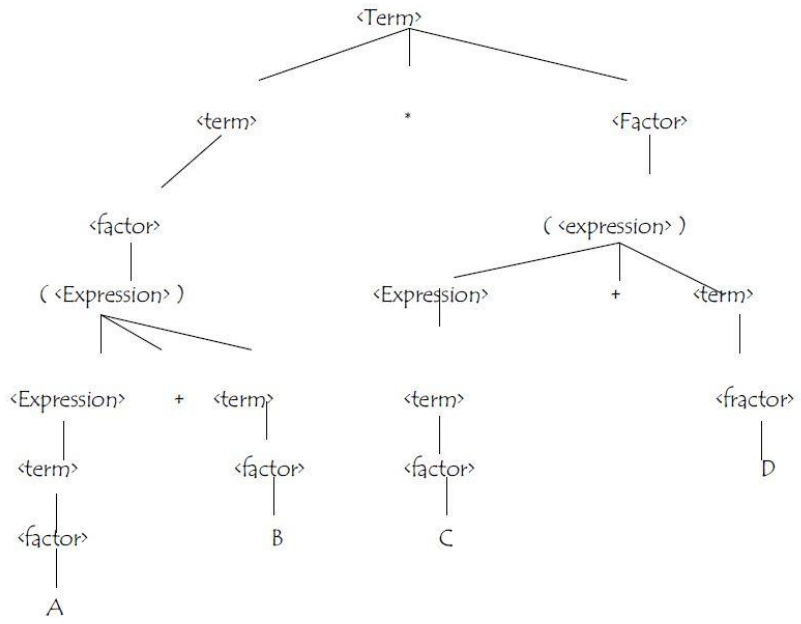


Gambar 2.12 Flowchart proses dari parsing

2.5.5. Pohon Sintaks

Sebuah tree adalah suatu graph terhubung yang tidak sirkuler dan memiliki satu buah root (akar) dan dari situ memiliki lintasan ke setiap simpul (daun/leaf). Parse Tree berfungsi untuk menggambarkan bagaimana memperoleh suatu string dengan cara menurunkan simbol-simbol variabel menjadi simbol-simbol terminal, sampai tidak ada simbol yang belum tergantikan.

Berikut ini adalah contoh tree yang dapat dilihat pada gambar 2.13 :



Gambar 2.13 Contoh *tree* atau Pohon

2.5.6. Text Mining

Text Mining, yang terkadang disebut "analisis teks" adalah salah satu cara untuk membuat data kualitatif atau "tidak terstruktur" yang dapat digunakan oleh komputer.pada studi *text mining*, dokumen umum nya digunakan sebagai unit dasar analisis. Sebuah dokume adalah urutan kata-kata dan tanda baca nya mengikuti aturan tata Bahasa (*grammar*) dari Bahasa yang berisi setiap segmen yang relevan dari teks dan dapat dari setiap panjangn kalimat. Pada hal ini semua yang dapat berisi kata seperti kertas, esai,buku, halaman web atau email dapat menjadi dokumen umum tergantung pada jenis analisis yang dilakukan.

Data teks yang ada akan diproses menjadi data numerik agar dapat dilakukan proses lebih lanjut. Sehingga dalam text mining terdapat istilah preprocessing data, yaitu proses pendahulu yang diterapkan terhadap data teks yang bertujuan untuk menghasilkan data numerik.

Pada proses preprosesing merupakan tahap dimana deskripsi di tangani untuk dapat siap diproses memasuki tahap text mining. Tahap-tahap ersebut adalah:

1) *Parsing/Tokenizing*
2) *Stopwords Removal/Filtering*
3) *Stemming*
4) *Tagging*
5) *Analyzing*

2.3. Chatbot

Penerapan dari AI (artificial intelligence) salah satunya dalam bentuk chat bot. Chat bot merupakan sebuah program komputer yang diprogram untuk dapat berkomunikasi dengan manusia menggunakan bahasa manusia itu sendiri.Salah satunya contoh kongkritnya adalah Help Bot pada Yahoo messenger dan ALICE (Artificial Linguistic Internet Computer Entity) yang dikembangkan oleh para peneliti bernama Dr. Richard S. Wallace.[13]

Atau dapat dikatakan juga chatbot (atau chatterbot, atau bots) adalah sebuah program komputer yang dirancang untuk menstimulasikan percakapan intelektual dengan satu atau lebih manusia baik secara audio maupun teks.

Pada mulanya, program komputer (bots) ini diuji melalui Turing Test, yaitu dengan merahasiakan identitasnya sebagai mesin sehingga dapat mengelabui orang yang bercakap-cakap dengannya. Jika pengguna tidak dapat mengidentifikasi bots sebagai suatu program komputer, maka chatterbot tersebut dikategorikan sebagai kecerdasan buatan (atau artificial intelligence). Zaman sekarang ini, chatterbot telah dimanfaatkan untuk tujuan praktis seperti bantuan online, layanan personal, atau akuisisi informasi, dalam hal ini dapat dilihat fungsi program sebagai suatu jenis agen percakapan (atau conversational agent). Yang membedakan chatterbot dengan sistem

pemrosesan bahasa alami (atau Natural Language Processing System) adalah kesederhanaan algoritma yang digunakan.

Meskipun banyak bots yang tampaknya dapat menginterpretasikan dan menanggapi masukan dari manusia, sebenarnya bots tersebut hanya menganalisis kata kunci dalam masukan dan membalasnya dengan kata kunci yang paling cocok, atau pola kata-kata yang paling mirip dari basis data tekstual.

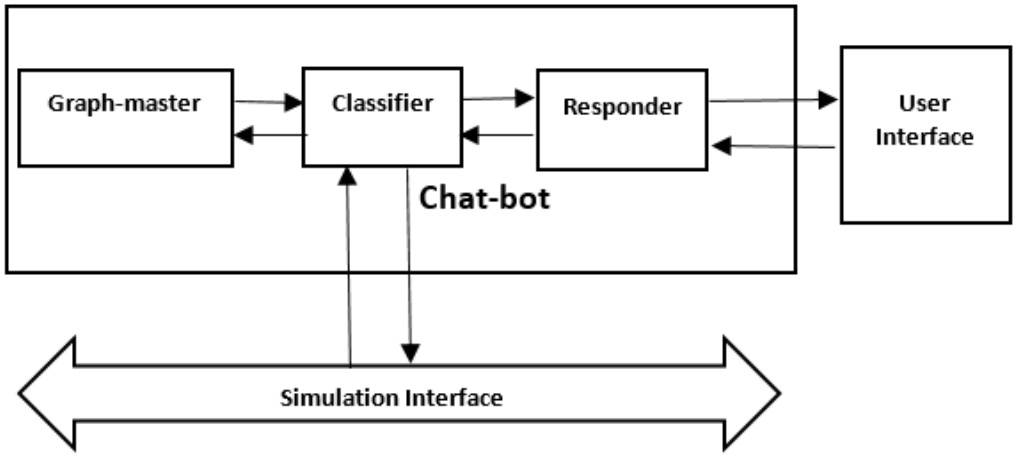
Chatbot juga merupakan QA system atau question-answering system, yaitu memberikan kemampuan pada sebuah mesin (komputer) untuk menginterpretasikan bahasa alami untuk melakukan dialog dengan pengguna hampir seperti dialog antara dua orang manusia dalam bahasa sehari-hari.

Baru-baru ini telah muncul ide-ide baru di dalam pengembangan ALICE di antaranya adalah kemampuan untuk menjaga alur pembicaraan untuk setiap user yang melakukan pembicaraan dengannya. Selain itu juga mulai telah dipikirkan suatu cara agar nantinya program tersebut dapat memprediksi atau mengidentifikasi umur, jenis kelamin,lokasi geografis dan pekerjaan dari client (user) yang melakukan perbincangan dengannya

2.3.1. Komponen Chatbot

Paket *software chat-bot* terdiri dari tiga komponen penting: *Responder*, *Classifier* dan *Graph master*. Antarmuka antara rutin utama dan pengguna adalah *Responder*. Responden mentransfer data dari pengguna ke *Classifier*. *Responder* mengontrol *input* dan *output*nya. Fungsi *Classifier* adalah menormalkan dan menyaring data masukan. Masukan pengguna diganti dan dibagi menjadi komponen logis oleh *Classifier*. *Classifier* memindahkan kalimat yang dinormalisasi ke dalam komponen *Graph-master* dari *chat-bot*. *Classifier* memproses *output* dari komponen *master Graph* dari *chat-bot*. *Classifier* juga menangani instruksi sintaks *database* (misalnya AIML). Komponen *graph-master chat-bot* menangani pola pencocokan. *Graph-master* bertanggung jawab untuk mengatur penyimpanan konten otak *chat-bot*.

Komponen *Graph-master* menyimpan isi otak sebagai grafik. *Graph-master* juga menangani proses pencocokan pola dan algoritma pencocokan pola. Diagram komponen *chat-bot* ditunjukkan pada Gambar di bawah ini.[14]



Gambar 2.13 Komponen Chatbot

2.3.2. Artificial Intelligent Markup Language (AIML)

AIML singkatan dari *Artificial Intelligence Markup Language*. Komunitas perangkat lunak bebas, Bahasa AIML menyederhanakan tugas pemodelan Bahasa Alami, sehubungan dengan proses "stimulus-respons" [15]. Objek AIML adalah tag bahasa. Objek AIML dikaitkan dengan perintah bahasa. Komponen dasar AIML disebut sebagai kategori. Setiap kategori adalah komponen struktural dari basis pengetahuan yang terdapat dalam chat-bot. Kategori terdiri dari: teks masukan pengguna, respons output terhadap teks masukan pengguna dan konteks pilihan. Struktur objek AIML adalah sebagai berikut :

<command> List of parameters </command>

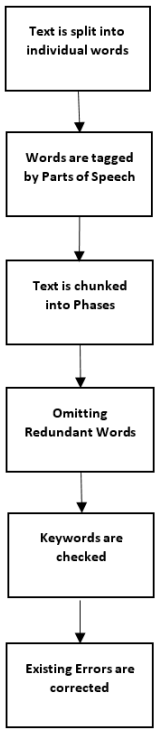
Struktur kategori AIML, pola, dan objek template adalah sebagai berikut:

```
<category>
        <pattern> User input text </pattern>
            <template>
                Output response to the user
```

```
                    </template>
</category>
```

2.3.2.1. Text Processing, Response dan Action

Input teks pengguna dipecah menjadi beberapa kata terpisah untuk menandai label *part-of-speech* sehubungan dengan posisi dan tetangga mereka dalam teks masukan. Pada tahap berikutnya, dengan bantuan berbagai jenis tata bahasa, kata-kata yang diberi label secara individual digabungkan untuk mengembangkan ungkapan. Pada tahap operasi chinking, kata kunci penting diambil dari frasa dengan menghapus kata-kata yang tidak diinginkan. Kata kunci diperiksa dan dikoreksi jika tidak benar. Tahap fase pemrosesan teks yang berbeda digambarkan pada gambar dibawah ini.



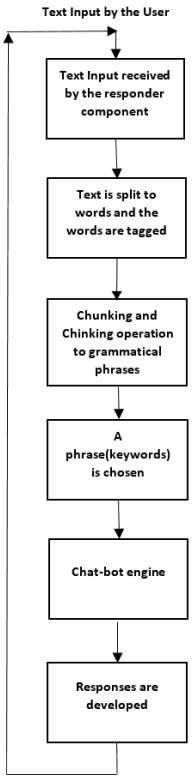
Gambar 2.14 Fase dari Text Processing

2.3.2.2. Text Processing

Chat-bot dapat dirancang untuk memberikan respons yang diharapkan terhadap percakapan teks bahasa manusia. Mesin *Chat-bot* dilengkapi dengan kata kunci yang diambil dari pemrosesan teks bahasa alami. *Output*nya adalah respon, yang akan ditunjukkan kepada pengguna. Diagram yang menggambarkan respons dan fase pengambilan tindakan.

2.3.2.3. Human Computer Conversation

Seorang manusia dan komputer bisa melakukan percakapan komputer manusia baik dengan mengetikkan teks atau dialog ucapan dengan menggunakan suara. Diagram yang menggambarkan tahap analisis dan pemrosesan penting untuk melakukan percakapan komputer ditunjukkan pada gambar



Gambar 2.15 Tahapan Human Komputer Conversation

2.4. *Dialogflow* API

Dialogflow.com adalah platform untuk mengembangkan chat-bots berdasarkan percakapan bahasa alami. Konsep penting seperti *Intents* and *Contexts* digunakan untuk memodelkan perilaku *chat-bot*. Maksudnya adalah pemetaan antara apa yang pengguna masukan dan respon atau tindakan apa yang harus dilakukan oleh *bot*. Konteks digunakan untuk membedakan input pengguna yang mungkin memiliki maksud yang berbeda tergantung pada input pengguna sebelumnya. Saat pengguna memasukkan data ke dalam platform Dialogflow.com, pertama kali diperiksa jika cocok dengan maksud yang telah ditentukan sebelumnya. Dialogflow.com memiliki fitur bernama "*Default fallback intent*" untuk menangani input pengguna yang tidak sesuai dengan maksud yang telah ditentukan sebelumnya. Kasus pencocokan suatu maksud dapat dibatasi dengan menyebutkan daftar konteks yang seharusnya bisa berjalan. Kasus pencocokan suatu maksud dapat membuat dan menghapus konteksnya. Sistem maksud dan konteks ini membuat ketentuan untuk mengembangkan *chat-bots* yang bisa memiliki arus besar dan kompleks. Keterbatasan Dialogflow.com adalah: *chat-bot* tidak dapat dirancang sedemikian rupa sehingga sebuah tujuan dapat dicocokkan hanya jika konteks tertentu tidak ada .

2.5. Perangkat Lunak

Perangkat Lunak (software) merupakan suatu program yang dibuat oleh pembuat program untuk menjalankan perangkat keras komputer.Perangkat Lunak adalah program yang berisi kumpulan instruksi untuk melakukan proses pengolahan data. Software sebagai penghubung antara manusia sebagai pengguna dengan perangkat keras komputer, berfungsi menerjemahkan bahasa manusia ke dalam bahasa mesin sehingga perangkat keras komputer memahami keinginan pengguna dan menjalankan instruksi yang diberikan dan selanjutnya memberikan hasil yang diinginkan oleh manusia tersebut.

Perangkat lunak terdiri dari kode-kode yang dibuat atau ditulis dalam bahasa pemrograman tertentu. Banyak bahasa pemrograman yang dapat dijadikan alat bantu

untuk merancang sebuah perangkat lunak antara lainbahasa PASCAL, C, C++, BASIC, Java, HTML, PHP, JavaScript, AIML, Dan bahkan tersedia tool yang berbasis grafis seperti Visual Basic, Visual C++, Borland Delphi, dan masih banyak lagi tool dan bahasa pemrograman lainnya.

2.6. *Object Oriented Programming* (OOP)

Object-Oriented Programming (OOP) adalah sebuah pendekatan untuk pengembangan / development suatu software dimana dalam struktur software tersebut didasarkan kepada interaksi object dalam penyelesaian suatu proses/tugas. Interaksi tersebut mengambil form dari pesan-pesan dan mengirimkannya kembali antar object tersebut. Object akan merespon pesan tersebut menjadi sebuah tindakan / *action* atau metode.

Object-oriented programs terdiri dari *objects* yang berinteraksi satu sama lainnya untuk menyelesaikan sebuah tugas. Seperti dunia nyata, *users* dari *software* programs dilibatkan dari logika proses untuk menyelesaikan tugas. Adapun karakteristik yang dimiliki OOP yaitu

a. *Objects*

   Object adalah sebuah *structure* yang menggabungkan data dan prosedur untuk bekerja bersama-sama.

b. *Abstraction*

   Ketika manusia berinteraksi dengan *object-object* di dunia ini, manusia sering hanya konsentrasi dengan sebuah bagian dari propertiesnya. Ketika membangun *objects* dalam aplikasi OOP, adalah penting untuk menggabungkan konsep *abstraction* ini. Jika membangun aplikasi *shipping*, maka harus membangun *object* produk dengan atribut seperti ukuran dan berat. Warna adalah contoh informasi yang tidak ada hubungannya dan harus dibuang. Tetapi ketika akan membangun *order-entry application*, warna menjadi penting dan harus termasuk atribut *object* produk.

c. *Encapsulation*

Ciri penting lainnya dari OOP adalah *encapsulation*. *Encapsulation* adalah sebuah proses dimana tidak ada akses langsung ke data yang diberikan, bahkan *hidden*. Jika ingin mendapat data, harus berinteraksi dengan *object* yang bertanggung jawab atas data tersebut.

d. *Polymorphism*

Polymorphisms adalah kemampuan 2 buah *object* yang berbeda untuk merespon pesan permintaan yang sama dalam suatu cara yang unik. Dalam OOP, diterapkan tipe *polymorphism* melalui proses yang disebut *overloading*. Dapat dilakukan dalam implementasi metode yang berbeda pada sebuah *object* yang mempunyai nama yang sama.

e. .*Inheritance*

Penggunaan *inheritance* dalam OOP untuk mengklasifikasikan *objects* dalam program sesuai karakteristik umum dan fungsinya.

2.7. Unified Model Language

UML *(Unified Modeling Language)* adalah sebuah bahasa untuk menetukan, visualisasi, kontruksi, dan mendokumentasikan *artifact* (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat lunak. Artifact dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya[17].

2.8.

UML merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan sistem yang besar dan kompleks. UML tidak hanya digunakan dalam proses pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan.

Bagian-bagian utama dari UML adalah view, diagram, model element, dan *general mechanism*.

1. *View*

   View digunakan untuk melihat sistem yang dimodelkan dari beberapa aspek yang berbeda. *View* bukan melihat grafik, tapi merupakan suatu abstraksi yang berisi sejumlah diagram. Beberapa jenis *view* dalam UMLantara lain: *use case view, logical view, component view, concurrency view, dan deployment view.*

   a. *Use case view*

      Mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai yang diinginkan *external actors*. *Actor* yang berinteraksi dengan sistem dapat berupa user atau sistem lainnya. *View* ini digambarkan dalam *use case diagrams* dan kadang-kadang dengan *activity diagrams*. *View* ini digunakan terutama untuk pelanggan, perancang (*designer*), pengembang (*developer*), dan penguji sistem (*tester*).

   b. *Logical view*

      Mendeskripsikan bagaimana fungsionalitas dari sistem, struktur statis (*class, object,dan relationship*) dan kolaborasi dinamis yang terjadi ketika *object* mengirim pesan ke *object* lain dalam suatu fungsi tertentu. *View* ini digambarkan dalam class diagrams untuk struktur statis dan dalam *state, sequence, collaboration, dan activity diagram* untuk model dinamisnya. View ini digunakan untuk perancang (*designer*) dan pengembang (*developer*).

   c. *Component view*

Mendeskripsikan implementasi dan ketergantungan modul. Komponen yang merupakan tipe lainnya dari code module diperlihatkan dengan struktur dan ketergantungannya juga alokasi sumber daya komponen dan informasi administrative lainnya. View ini digambarkan dalam *component view* dan digunakan untuk pengembang (*developer*).

d. Concurrency view

Membagi sistem ke dalam proses dan prosesor. *View* ini digambarkan dalam diagram dinamis *(state, sequence, collaboration, dan activity diagrams)* dan diagram implementasi *(component dan deployment diagrams)* serta digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

e. Deployment view

Mendeskripsikan fisik dari sistem seperti komputer dan perangkat (*nodes*) dan bagaimana hubungannya dengan lainnya. *View* ini digambarkan dalam *deployment diagrams* dan digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

2. Diagram

Diagram berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu *view* tertentu dan ketika digambarkan biasanya dialokasikan untuk *view* tertentu. Adapun jenis diagram antara lain :

a. Use Case Diagram

Use case adalah abstraksi dari interaksi antara sistem dan actor. *Use case* bekerja dengan cara mendeskripsikan tipe interaksi antara *user* sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. *Use case*

merupakan konstruksi untuk mendeskripsikan bagaimana sistem akan terlihat di mata *user*. Sedangkan *use case* diagram memfasilitasi komunikasi diantara analis dan pengguna serta antara analis dan *client*. dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem

b. Class Diagram

Class adalah dekripsi kelompok obyek-obyek dengan *property*, perilaku (operasi) dan relasi yang sama. Sehingga dengan adanya *class* diagram dapat memberikan pandangan global atas sebuah sistem. Hal tersebut tercermin dari class- class yang ada. Sebuah sistem biasanya mempunyai beberapa *class* diagram. *Class* diagram sangat membantu dalam visualisasi struktur kelas dari suatu sistem

c. Component Diagram

Component software merupakan bagian fisik dari sebuah sistem, karena menetap di komputer tidak berada di benak para analis. Komponen merupakan implementasi *software* dari sebuah atau lebih *class*. Komponen dapat berupa *source code*, komponen biner, atau *executable component*. Sebuah komponen berisi informasi tentang *logic class* atau *class* yang diimplementasikan sehingga membuat pemetaan dari *logical view* ke *component view*. Sehingga component diagram merepresentasikan dunia nyata yaitu *component software* yang mengandung *component, interface dan relationship*.

d. Deployment Diagram

Menggambarkan tata letak sebuah sistem secara fisik, menampakkan bagian-bagian *software* yang berjalan pada bagian-bagian *hardware*, menunjukkan hubungan komputer dengan perangkat (*nodes*) satu sama lain dan jenis hubungannya. Di dalam *nodes*, *executeable component* dan *object* yang dialokasikan untuk memperlihatkan unit perangkat lunak yang dieksekusi oleh *node* tertentu dan ketergantungan komponen.

e. State Diagram

Menggambarkan semua *state* (kondisi) yang dimiliki oleh suatu *object* dari suatu *class* dan keadaan yang menyebabkan *state* berubah. Kejadian dapat berupa *object* lain yang mengirim pesan. *State class* tidak digambarkan untuk semua *class*, hanya yang mempunyai sejumlah *state* yang terdefinisi dengan baik dan kondisi *class* berubah oleh *state* yang berbeda.

f. Sequence Diagram

Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah *scenario*. Kegunaannya untuk menunjukkan rangkaian pesan yang dikirim antara *object* juga interaksi antara *object*, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.

g. Collaboration Diagram

Menggambarkan kolaborasi dinamis seperti *sequence* diagrams. Dalam menunjukkan pertukaran pesan, *collaboration* diagrams menggambarkan *object* dan hubungannya (mengacu ke konteks). Jika penekannya pada waktu atau urutan gunakan

sequence diagrams, tapi jika penekanannya pada konteks gunakan *collaboration* diagram.

h. Activity Diagram

Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi.