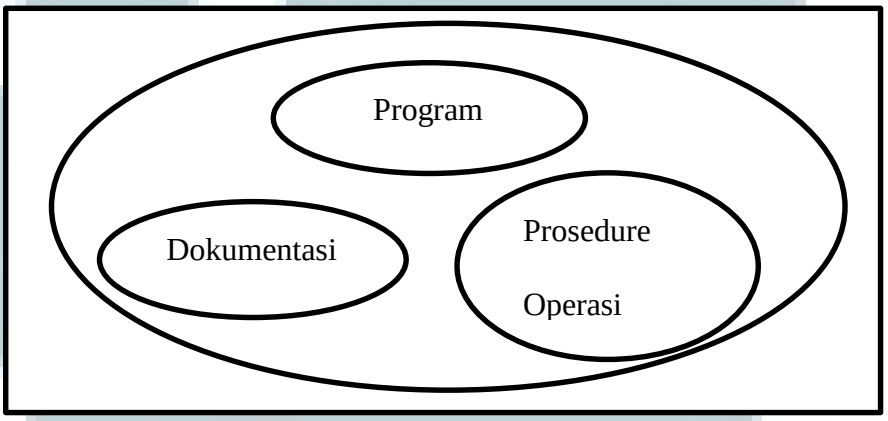
BAB II

TINJAUAN PUSAKA

2.1 Piranti Lunak

Yang disebut dengan perangkat lunak mencakup program komputer, dokumentasi program tersebut, konfigurasi yang diperlukan sehingga program tersebut dapat berjalan (Sommerville, 2011).



Gambar 2.1 Komponen Piranti Lunak

Perangkat lunak dapat dibagi menjadi dua tipe yaitu sebagai berikut.

1. Produk Generik

Merupakan produk *standalone* (berdiri sendiri) yang dijual oleh suatu organisasi pengembang dan dijual ke pasar terbuka agar dapat bisa dibeli siapapun yang memerlukannya. Contohnya. program pengolah kata, paket untuk menggambar dan alat bantu proyek manajemen.

2. Produk Pesanan

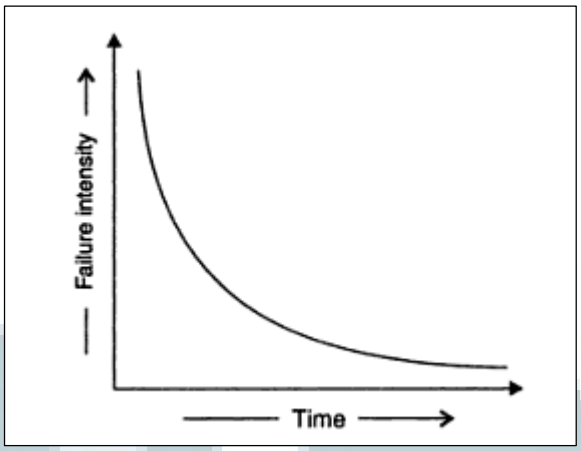
Merpakan sistem-sistem tertentu yang dipesan oleh pelanggan. Perangkat lunak ini dikembangkan khusus untuk pelanggan tersebut. Contohnya. sistem

kontrol untuk piranti elektronik, sistem yang ditulis untuk bisnis tertentu, sistem kontrol lalu lintas udara.

Piranti lunak mempunyai karakteristik yang membedakannya dengan produk-produk lain yang dihasilkan manusia seperti perangkat keras komputer, beberapa karakteristik yang penting adalah sebagai berikut (Aggarwal dan Singh, 2001).

1. *Software does not wear out*

Software atau piranti lunak mempunyai karakteristik semakin lama suatu piranti lunak digunakan, piranti lunak semakin menjadi dapat dipercaya. Hal ini dikarenakan semakin lama suatu *software* dipakai maka software tersebut akan menerima lebih banyak *update* yang akan dapat semakin memaksimalkan dan menutupi kelemahan dari *software* tersebut. Berbeda dengan *hardware* komputer ada 3 fase hidup yang dikenal yaitu fase pertama atau awal yang disebut dengan *burn-in-phase,* pada tahap ini di ekspektasikan banyak terjadi kegagalan. Fase kedua adalah fase diekspektasikannya kegunaan produk tersebut. Fase ketiga adalah yang disebut *wear out phase* yaitu tahap dimana diekspektasikan banyak kegagalan yang akan dialamai. Hal ini diakibatkat karena suatu *hardware* akan keadaannya akan terpengaruh oleh pengaruh lingkungan. Hal ini sangat berbeda dengan *software* yang tidak akan terpengaruh oleh faktor lingkungan. Gambar 2.2 menunjukan perbandingan tingkat kesalahan yang dialami suatu *software* terhadap waktu.



Gambar 2.2 Software Curve
Sumber : Aggarwal dan Singh (2001)

Dapat dilihat pada grafik di atas bahwa semakin lama umur dari suatu *software* maka *software* tersebut akan semakin dapat diandalkan hal ini disebabkan *software* dapat menerima perbaikan-perbaikan yang dibuat oleh *developer* dari *software* tersebut untuk memperbaiki kelemahan-kelemahan yang dimiliki *software*.

2. Software is not manufactured

Jika suatu perangkat lunak diproduksi secara banyak maka hal tersebut tidak akan meningkatkan biaya produksi melainkan meingkatkan biaya perawatan atau *maintenance.* Sedangkan jika suatu perangkat keras diproduksi secara banyak maka hal tersebut akan meningkatkan biaya produksi perangkat keras tersebut.

3. Reusability of component

Seiring dengan perkembangan teknologi, sekumpulan standart baru akan muncul dan diciptakan oleh para ilmuwan. Piranti lunak juga mengikuti aturan yang sama dengan hal yang sama yaitu aplikasi juga akan berkembang secara terus menerus dan menciptakan suatu standart yang baru tetapi aplikasi memiliki jangka waktu hidup komponen yang lebih lama bila dibandingkan dengan komponen dari

perangkat keras dan juga suatu aplikasi dapat dimodifikasi untuk dapat mengikuti standart yang baru. Oleh karena itu dapat disimpulkan bahwa suatu piranti lunak memiliki tingkat *reusability* yang tinggi.

2.2 Rekayasa Piranti Lunak

Yang dimaksud dengan rekayasa piranti lunak adalah metode, alat alat dan teknik yang digunakan untuk mengembangkan suatu *software* (Bell, 2005). Perbedaan antara rekayasa piranti lunak dengan *computer science* adalah rekayasa piranti lunak berhubungan dengan masalah-masalah dalam memproduksi sebuah perangkat lunak sedangkan *computer science* membahas teori dan metode yang mendasari sistem komputer.

2.3 *Grammar*

Kata grammar diturunkan dari bahasa Yunani yang berari seni dari huruf yang dimana huruf itu diartikan sebagai melukis atau menulis (Harper). Grammar atau tata bahasa merupakam suatu aturan-aturan yang mengatur penggunaan dari suatu bahasa. Aturan-aturan dari bahasa itu didapat dari hasil penerjemahan pengembangan bahasa tersebut dan juga berasal dari hasil observarsi dan dokumentasi bahasa tersebut.

2.4 Grammar Checker

Grammar checker merupakan suatu aplikasi yang melakukan pendeteksian atas kesalahan yang terjadi saat penulisan atau penggunaan bahasa. Proses dari

suatu aplikasi *grammar correction* mempunya 4 langkah kerja adalah sebagai berikut (Sågvall, 1998).

1. Pengidentifikasian bagian-bagian yamg mempunyai kesalahan *grammar*
2. Pengidentifikasian bagian- bagian yang melanggar aturan
3. Pengidentifikasian kemungkinan penyebab kesalahan
4. Pembuatan dan penyusunan kemungkinan perbaikan yang dapat dilakukan

Dalam menentukan kesalahan pada *grammar* kita dapat menggunakan 2 cara yaitu dengan cara *positive grammar* atau *negative grammar*. *Positive grammar* bekerja dengan menghasilkan kalimat kalimat yang baku. *Negative Grammar* bekerja dengan mendeskripsikan kesalahan kesalahan yang dapat terjadi (Sågvall, 1998).

2.5 Algoritma *Minimal Correction*

Algoritma *minimal correction* merupakan suatu algoritma yang meminimalisasi perbaikan yang dilakukan pada suatu kalimat sehingga akan menghasilkan hasil yang lebih benar, hal tersebut dilakukan dengan melakukan memberikan biaya pada setiap perubahan yang dilakukan pada suatu kalimat dan dan memilih pilihan perbaikan yang menggunakan biaya yang paling sedikit(Clement *et al.*, 2009).

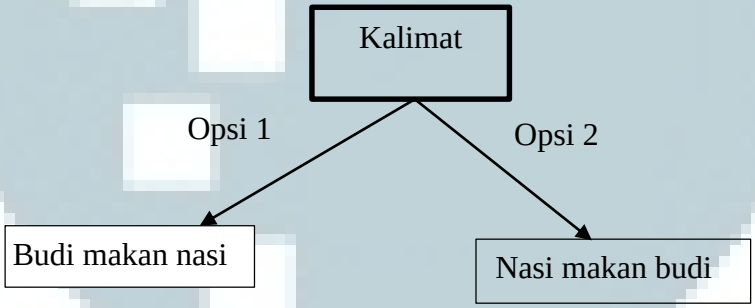
Biaya yang dilakukan pada suatu kalimat dapat direpresentasikan sebagai tugas yang akan dilakukan oleh aplikasi dan dapat direpresentasikan dengan rumus sebagai berikut.

$$\propto = \left(u_{\varphi}\right)_{\varphi \epsilon \phi}$$

Rumus 2.1 *Feature Assigmet Cost*

Dimana ϕ adalah perubahan-perubahan yang dapat dilakukan pada suatu kalimat yang dapat diberi suatu nilai. Dan $(u_\varphi)_\varphi$ adalah gabungan dari biaya-biaya perubahan yang digunakan untuk melakukan perbaikan pada kalimat. Dan $\propto$ adalah total biaya yang diperlukan untuk melakukan perbaikan pada suatu kalimat. Contoh dari penggunaan rumus itu adalah sebagai berikut.

Jika diberikan suatu kalimat : makan budi nasi maka perhitungan dan perbandingan biaya yang dilakukan untuk melakukan perbaikan pada kalimat tersebut adalah sebagai berikut.



Jika menggunakan pilihan opsi satu maka biaya yang digunakan untuk melakukan perubahan tersebut adalah satu dikarenakan perubahan yang dilakukan pada kalimat asal hanya satu kali maka berdasarkan rumus 2.1 maka dapat didapatkan total biaya atau *feature assignment cost* sebanyak satu dari tiga perbaikan yang dapat dilakukan.

Jika menggunakan pilihan opsi dua maka biaya yang digunakan untuk melakukan perubahan tersebut adalah dua dikarenakan terjadi perubahan sebanyak dua kali dari kalimat asal yang diberikan, maka berdasarkan rumus 2.1 maka dapat didapatkan total biaya atau *feature assignment cost* sebanyak dua dari tiga perbaikan yang dapat dilakukan.

Sehingga algoritma minimal correction ini akan memilih untuk melakukan perbaikan yang disarankan oleh opsi satu yang hanya menggunakan satu *feature assignment cost* dibandingkan opsi dua yang menggunakan dua *feature assignment cost*.

2.6 Algoritma *Deep Parsing*

Full parsing atau *deep parsing* menganalisa struktur dari sebuah kalimat dan mengekstrak informasi yang berguna (Tsuruoka *et al.*, 2009). *Full Parsing* atau *deep parsing* adalah proses *parsing* yang berbasiskan *formal grammar* yang terdiri dari HPSG (*Head Driven Phrase Structure*), LFG (*Lexical Function Grammar*), dan CFG (*Context Fre Grammar*). *Deep Parsing* menggunakan suatu metode pencarian dan formula grammar khusus (Head Driven Phrase Structure, Lexical Function Grammar, Context Fre Grammar) untuk mendapatkan struktur sintaks suatu kalimat. Untuk mendapatkan sintaks kalimat tersebut kalimat akan dibagi bagi menjadi beberapa subset. Salah satu bentuk dari *grammar* yang bisa digunakan adalah CFG (*Context Free Grammar*).

CFG atau *Context Free Grammar* dikembangkan oleh Noam Chomsky. *Context Free Grammar* atau yang bisa disebut juga *phrase-structure grammar* merupakan suatu sintaks yang di dapat dari *natural language* yang dideskripsikan oleh *context free rules* yang dikombinasikan dengan *transformation rules* (Hopcroft, 1979). *Context Free Grammar* memberikan bentuk matematik dan spesifik atas bentuk struktur dari suatu *natural language.*

Context free Grammar dapat dideskripsikan sebagai berikut (Hopcroft, 1979).

$$G = (V, \Sigma, R, S)$$

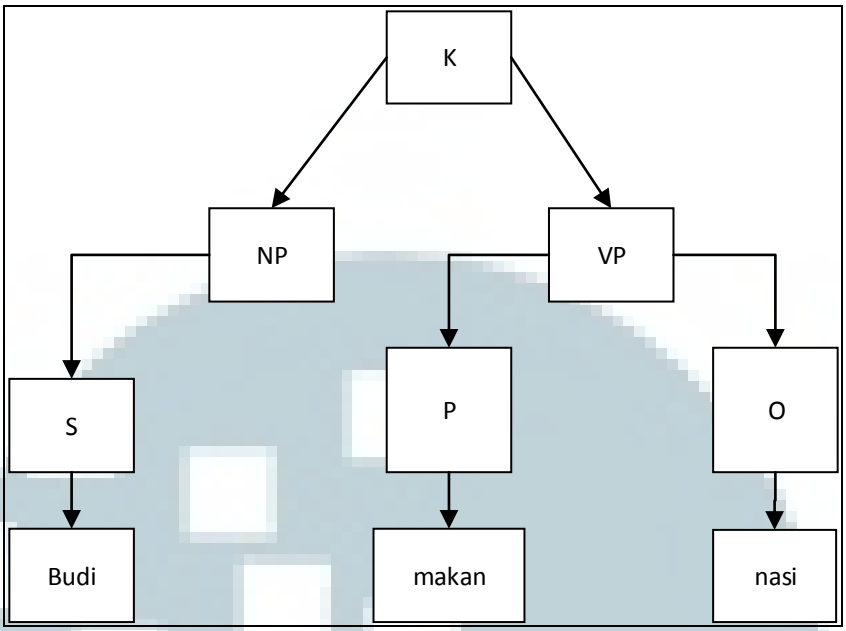
Rumus 2.2 *Context Free Grammar*

Dimana V merupakasn sebuah *finite set* yang setiap $v \epsilon V$ adalah variable dimana setiap variabelnya merepresentasikan suatu kata dalam sebuah kalimat. Dan Σ merupakan suatu *finiter set of terminal* yang merupakan uraian dari V. R merupakan peraturan produksi dari tata bahasa yang digunakan. Dan S merupakan *start variable* yang digunakan untuk merepresentasikan kalimat awal.

Contoh dari algoritma *deep parsing* yang menggunakan CFG adalah sebagai berikut.

Kalimat : Budi makan nasi.

Dengan mengimplementasikan algoritma *deep parsing* yang menggunakan CFG (*Context Free Grammar*) maka dapat kalimat tersebut dapat dibagi-bagi menjadi sebagai berikut.

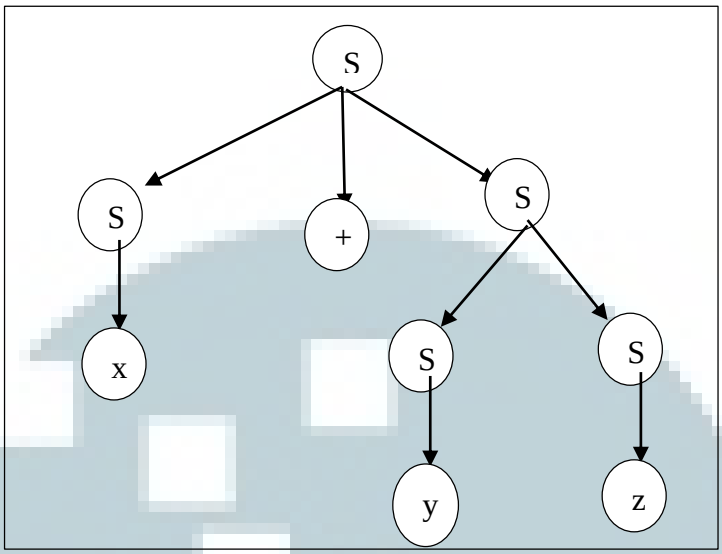


Gambar 2.3 Bagan Cara Kerja Algoritma *Deep Parsing*

Keterangan :

- K : Kalimat
- NP : frase kata benda
- VP : frase kata kerja
- S : Subyek
- P : Predikat
- O : Obyek

Sehingga apabila contoh dari gambar 2.3 dirubah dalam bentuk *context free grammar* maka akan didapat bagan *parse tree* sebagai berikut.



Gambar 2.4 Parse Tree Context Free Grammar

Dengan *state* awal (S) adalah Budi makan nasi. Dan dengan target yaitu x yang bernilai Budi, y yang bernilai makan dan z yang bernilai nasi.

Algoritma *deep parsing* ini memiliki keuntungan yaitu mempunyai bentuk matematika yang baik, algoritma yang efisien, dan baik untuk digunakan untuk bahasa yang mempunyai stuktur kalimat yang pasti (Clement, 2009).

2.7 Bahasa Indonesia

Bahasa Indonesia lahir pada tanggal 28 Oktober 1928, yaitu pada saat dilaksanakannya sumpah pemuda. Bahasa Indonesia lalu diresmikan sebagai bahasa nasional negara Indonesia pada tanggal 18 Agustus 1945. Pada kongres bahasa yang dilaksanakan pada tahun 1954 dinyatakan bahasa Indonesia merupakan perkembangan dari bahasa Melayu (Kemendikbud, 2011). Lalu pada tanggal 12 Oktober 1972 Menteri Pendidikan dan Kebudayaan meyusun pedoman umum yang merupakan pemaparan ejaan. Pedoman yang dibuat itu berisi pedoman

pedoman untuk penggunan huruf, pemenggalan kata, pemakaian huruf kapital dan huruf miring, penulisan kata, singkatan dan akronim, penulisan unsur serapan dan pemakaian tanda baca (Tim Pusat Bahasa, 2000).