

BAB III METODE PENELITIAN

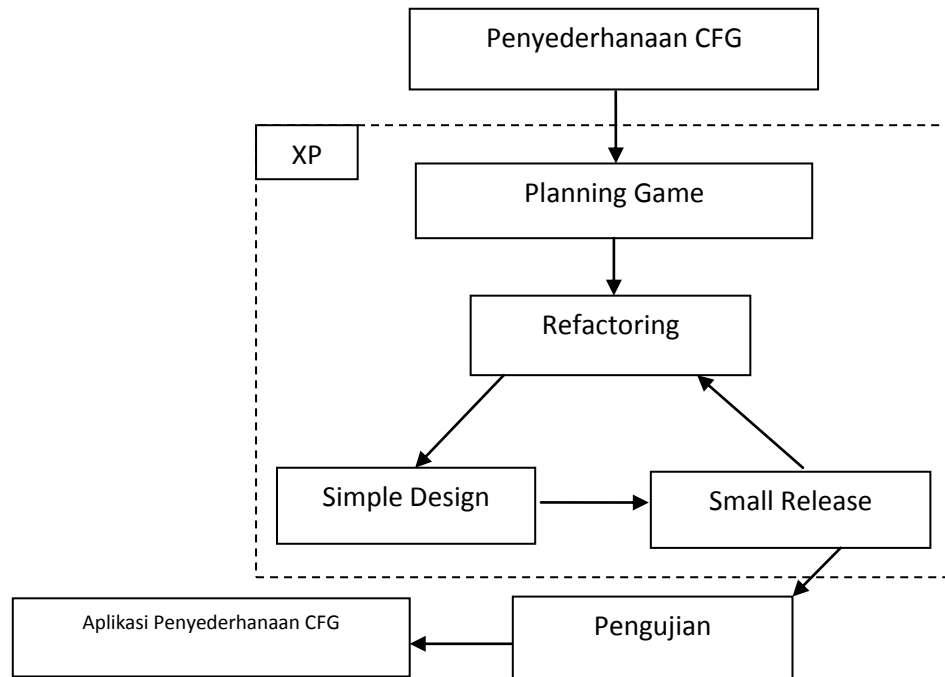
3.1 Waktu dan Tempat Penelitian

Penelitian ini dilakukan di Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung. Waktu penelitian dilakukan pada Semester Genap Tahun Ajaran 2013-2014.

3.2 Metode Pengembangan Sistem

Aplikasi Penyederhanaan *Grammar* ini dikembangkan menggunakan metode *Extreme Programming* (XP). Metode XP ini merupakan metode yang sangat fleksibel dan cocok dipakai untuk mengembangkan software dalam waktu singkat. Aplikasi Penyederhanaan *Context Free Grammar* ini hanya dikembangkan dalam waktu singkat dan sedikit *resource* serta masih berupa *prototype* yang setiap saat bisa saja ditambah atau dirubah *sourcenya*, sehingga penggunaan metode XP sangat membantu dalam pengembangan aplikasi ini.

Metode XP, seperti sudah dijelaskan pada bab tinjauan pustaka memiliki dua belas tahapan. Penelitian ini tidak menggunakan seluruh tahapan dalam XP karena aplikasi yang dikembangkan merupakan aplikasi kecil dan tim hanya terdiri dari satu *programmer*. Tahapan-tahapan XP yang digunakan dalam pembuatan Aplikasi Penyederhanaan *Grammar Context Free Grammar* adalah sebagai berikut:



Gambar 2. Diagram Metode Pengembangan sistem

3.2.1 The Planning Game

Planning game merupakan tahap menentukan kebutuhan aplikasi. Pengembang bertemu dengan *client* dan membahas seperti apa aplikasi yang diinginkan oleh *client* pada tahapan ini. Tahap ini dilakukan secara intensif hingga pengembang dan *client* memiliki pandangan yang sama tentang aplikasi yang dibuat. Pembuatan algoritma untuk menyederhanakan *Context Free Grammar* merupakan salah satu proses yang dikerjakan dalam tahapan *planning game*.

3.2.1.1 Algoritma penyederhaan *Context Free Grammar* (CFG)

Aplikasi yang dibuat merupakan aplikasi yang dipakai untuk menyederhanakan *Context Free Grammar*. Proses yang terjadi dalam menyederhanakan CFG adalah sebagai berikut.

L1. Masukkan jumlah produksi.

L2. Tampilkan field untuk menginput produksi sesuai jumlah yang dimasukkan *user*.

L3. Masukkan produksi.

L4. Cek produksi tersebut apakah mengandung produksi *epsilon*, jika tidak ke L6, jika iya ke L5.

L5. Hilangkan produksi *epsilon*.

Contoh. $A \Rightarrow AB, B \Rightarrow b \mid \wedge$ menjadi $A \Rightarrow AB \mid A, B \Rightarrow b. B \Rightarrow \wedge$ dihilangkan karena produksi *epsilon*

L6. Cek produksi tersebut apakah mengandung produksi *unit*. Jika tidak ke L8, jika iya ke L7.

L7. Substitusikan dengan produksi turunannya yang sama. Contoh. $A \Rightarrow B, B \Rightarrow c$ menjadi $A \Rightarrow c, B \Rightarrow c$. $A \Rightarrow B$ disubstitusikan menjadi $A \Rightarrow c$.

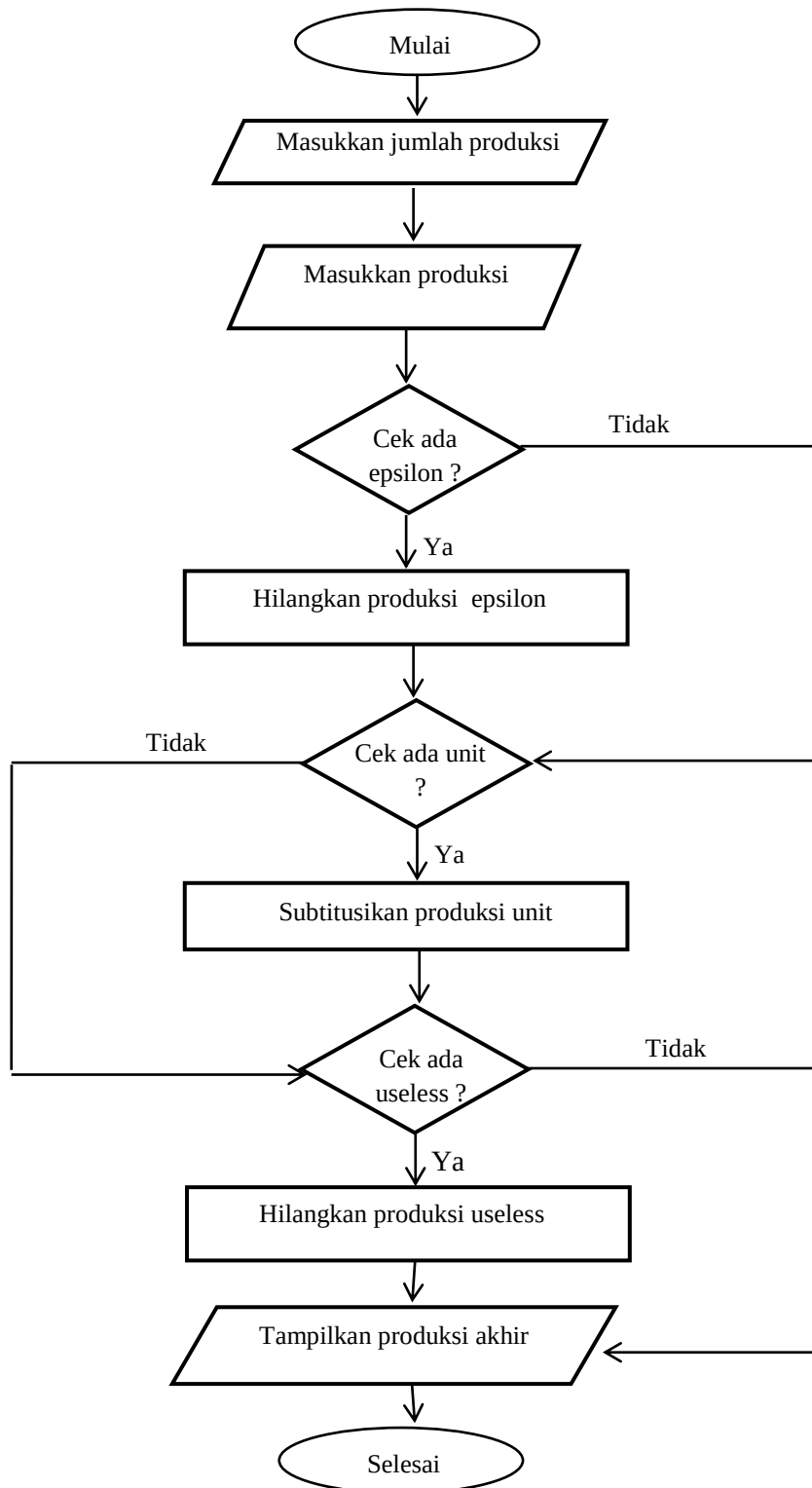
L8. Cek produksi tersebut apakah mengandung produksi *useless*. Jika tidak ke L10. Jika iya ke L9.

L9. Hilangkan produksi *useless*.

Contoh. $S \Rightarrow AbC \mid AB, A \Rightarrow a, B \Rightarrow b$ menjadi $S \Rightarrow AB, A \Rightarrow a, B \Rightarrow b$. $S \Rightarrow AbC$ dihilangkan karena produksi C *useless*.

L10. Tampilkan produksi yang tidak mengandung produksi *epsilon*, *unit* dan *useless* sebagai hasil akhir perhitungan.

Pembuatan Aplikasi Penyederhanaan *Context Free Grammar* digambarkan menggunakan *flowchart* yang dapat dilihat pada Gambar 3 berikut:



Gambar 3. *Flowchart* Penyederhanaan *Context Free Grammar*

3.2.2 Refactoring

Proses penulisan kode dalam membuat aplikasi penyederhanaan *grammar* ini menggunakan proses refactoring. Proses *refactoring* adalah proses untuk meningkatkan cara penulisan program namun tidak merubah proses dari program tersebut. Contoh *refactoring* ditunjukkan pada Kode Program 1 berikut:

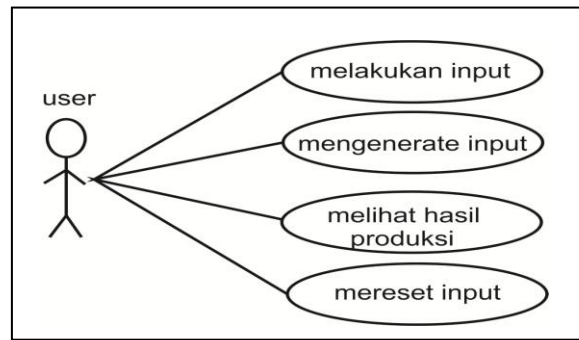
```
function show_array(){ //menampilkan hasil untuk UI
    global $nt, $produksi;
    for($i=0;$i<count($nt);$i++){
        echo"$nt[$i]=>";
        echo $produksi[$nt[$i]][0];
        for($j=1;$j<count($produksi[$nt[$i]]);$j++)
            echo " | ".$produksi[$nt[$i]][$j];
        echo"<br>";
    }
}
```

Kode Program 1. Ilustrasi *Refactoring*

Kode Program 1 menunjukkan aplikasi menggunakan *function* untuk membuat baris program terlihat rapih dan dapat dimengerti. Penggunaan *function* juga menguntungkan apabila terjadi *error*. Pendeteksian *error* akan lebih mudah sehingga bila ingin merubah *script* program tidak harus menulis dari awal.

3.2.3 Simple Design

Tahap ini merupakan tahap pembuatan desain. Desain yang dibuat untuk aplikasi ini berupa desain sederhana dimana semua *button* diberi nama dengan jelas sehingga memudahkan *client* dalam menggunakannya. Aplikasi ini pada dasarnya digunakan oleh banyak pihak karena aplikasi ini dihosting ke dalam laman *web*. Desain yang simpel membuat pengguna mudah mengerti dalam menjalankan aplikasi ini. Diagram *usecase* digunakan untuk membuat ilustrasi yang dilakukan pengguna di dalam aplikasi sehingga dapat mempermudah pengembang dalam membuatnya. Diagram *usecase* dapat dilihat pada Gambar 4 berikut:

Gambar 4. Diagram *Usecase*

3.2.3.1 *Form Awal*

Form ini berisi satu *textfield* dan tombol *submit*. Halaman ini dipakai untuk menentukan jumlah produksi yang diproses. *Form* awal dapat dilihat pada Gambar 5 berikut:

form masukan jumlah produksi	
masukan jumlah produksi submit	
footer	

Gambar 5. Ilustrasi *Form Awal*

3.2.3.2 *Form Masukan Produksi*

Form ini berisi *textfield* sesuai dengan jumlah masukan di halaman awal dan dua *button* yaitu *generate* dan *reset*. *Form* masukan produksi ditunjukkan pada Gambar 6 berikut:

form masukan produksi		
masukan produksi		
	→	
	→	
	→	
footer		

Gambar 6. Ilustrasi *Form Masukan Produksi*

3.2.3.3 *Form Hasil*

Form hasil hanya berupa laman kosong yang menampilkan hasil dari penyederhanaan dan produksi mana saja yang dibuang. *Form* hasil dapat dilihat pada Gambar 7 berikut:

form hasil penyederhanaan	
footer	

Gambar 7. Ilustrasi *Form Hasil*

3.2.4 *Small Release*

Tahap ini merupakan tahap dimana setiap pengembang menyelesaikan suatu unit proses dalam aplikasi dan akan langsung menunjukkannya kepada *client*. Aplikasi yang dibuat dalam penelitian ini memiliki banyak tahapan, seperti

saat penyederhanaan CFG ada tiga proses yaitu penghilangan *epsilon*, penghilangan *unit* dan penghilangan *useless*. Setiap proses itu selesai, pengembang menunjukan kepada *client* sehingga bila terdapat kekurangan dari proses tersebut dapat segera diatasi.

3.2.5 Testing

Sistem yang telah selesai dibangun melalui tahap pengkodean, kemudian diuji. Pengujian sistem dilakukan langsung kepada pengguna untuk mengetahui apakah sistem yang dibuat telah memenuhi kebutuhan pengguna atau belum. Pengujian sistem menggunakan metode *black box testing*. Pengujian *black box testing* berfokus pada *output* dari berbagai jenis *input*. *Black box testing* melibatkan pengujian *interface* untuk memastikan bahwa kode tersebut memenuhi persyaratan fungsional dan berfungsi. Sistem dikatakan berfungsi dengan baik apabila *input* diberikan sehingga sistem memberikan *output* sesuai dengan identifikasi sistem yang telah ditentukan sebelumnya

3.3 Perangkat Penelitian

Pengembangan aplikasi ini menggunakan satu unit *personal computer* (PC) dengan spesifikasi sebagai berikut:

1. Hardware

Spesifikasi hardware yang digunakan dalam pengembangan sistem ini adalah sebagai berikut:

1. Intel(R) Pentium(R) Dual CPU E2140 @1.60 GHz
2. Ram 1536MB

2. *Software*

Spesifikasi software yang digunakan adalah sebagai berikut:

1. *Operating System Windows Seven Ultimate 32-bit (6.1, build 7601)*
2. *Adobe Dreamweaver Portable Version*
3. *Notepad++*
4. XAMPP