3. DESAIN SISTEM

3.1. Desain *Stack* Pada Program

Dalam program pengecekan validasi argumen berbahasa Inggris diperlukan *stack* untuk melakukan perhitungan terhadap *sentencial calculus* yang dihasilkan dari proses translasi. Berikut ini merupakan desain tipe *stack* yang akan digunakan dalam program :

```
StackElementType = String[5];
StackEvalElementType = String;
StackArray = array [1..StackLimit] of StackElementType;
StackEvalArray = array [1..StackLimit] of StackEvalElementType;

StackType = record
  elemen : StackArray;
  top : 0..StackLimit;
end;

StackEvalType = record
  elemen : StackEvalArray;
  top : 0..StackLimit;
end;
```

3.2. Desain Tipe Data Pada Program

Dalam program pengecekan validasi argumen berbahasa Inggris diperlukan beberapa tipe data untuk menampung argumen yang di-*input*-kan oleh *user* dalam program. Berikut ini desain beberapa tipe data yang akan digunakan untuk menampung argumen dalam program :

a. Tipe data TKata.

   Tipe data TKata merupakan *record* dari *field* teks, status_proses, status_teks, dan huruf. Berikut ini merupakan deklarasi tipe data TKata :

```
TKata = record
      teks : String;
      status_proses : boolean;
      status_teks : char;
      huruf : char;
end;
```

Field teks bertipe *string*, merupakan variabel yang digunakan untuk menyimpan kata. *Field* status_proses bertipe *boolean*. *Field* status_proses akan bernilai *false* jika kata yang bersangkutan belum diproses, dan akan bernilai *true* jika kata yang bersangkutan sudah diproses. *Field* huruf bertipe *char*, merupakan variabel yang digunakan untuk menyimpan huruf translasi (proposisi) milik teks tersebut. *Field* status_teks bertipe *char*. *Field* status_teks digunakan untuk menyimpan status dari teks, dan status tersebut direpresentasikan dalam bentuk huruf. Bila *field* status_teks berisi huruf "N" maka berarti teks tersebut merupakan kata ingkaran. Bila *field* status_teks berisi huruf "V" maka berarti teks tersebut merupakan *verb* (kata kerja). Pada Tabel 3.1 dapat dilihat tabel huruf status teks dan representasinya, yang digunakan dalam program :

Tabel 3.1. Tabel Huruf Status Teks dan Representasinya.

Huruf	Representasi
N	Merupakan kata ingkaran
K	Merupakan kata konjungsi
D	Merupakan kata disjungsi
I	Merupakan kata implikasi
B	Merupakan kata biimplikasi
S	Merupakan kata *subject*
M	Merupakan kata *modal*
F	Merupakan kata *frequency*
V	Merupakan kata *verb*
O	Merupakan kata *object*
P	Merupakan kata *preposition*
C	Merupakan kata kesimpulan
X	Kata yang belum diolah atau kata yang bukan *keyword*
,	Tanda koma

b. Tipe data TKalimat.

Tipe data TKalimat merupakan *record* dari *field* kata, dan panjang_kalimat. Berikut ini merupakan deklarasi tipe data TKalimat :

```
TKalimat = record
      kata : Array [1..50] of TKata;
      panjang_kalimat : Integer;
end;
```

Field kata merupakan *array* dari TKata yang sudah didefinisikan di atas. Sedangkan *field* panjang_kalimat bertipe *integer*, merupakan suatu variabel untuk menyimpan panjang dari kalimat, yang diperoleh dari jumlah kata.

c. Tipe data TTranslasi.

Tipe data TTranslasi merupakan *record* dari *field* kalimat_ke, awal, akhir, huruf, ingkaran, subject, tobe, freq, verb, obj, dan prep. Berikut ini merupakan deklarasi tipe data TTranslasi :

```
TTranslasi = record
      Kalimat_ke : Integer;
      awal : Integer;
      akhir : Integer;
      huruf : Char;
      ingkaran : Boolean;
      subject : String;
      tobe : String;
      freq : String;
      verb : String;
      obj : String;
      prep : String;
end;
```

Field kalimat_ke bertipe *integer*, merupakan variabel untuk menyimpan kalimat ke-n yang ditranslasi. *Field* awal dan akhir bertipe *integer*, merupakan variabel untuk menyimpan batas awal kalimat dan batas akhir kalimat yang ditranslasi. *Field* huruf bertipe *char*, merupakan variabel untuk menyimpan

huruf translasi (proposisi) dari kalimat yang bersangkutan. *Field* ingkaran bertipe *boolean*. *Field* ingkaran akan bernilai *true* apabila kalimat merupakan kalimat ingkaran, dan akan bernilai *false*, apabila kalimat bukan merupakan kalimat ingkaran. *Field subject*, *tobe*, *freq*, *verb*, *obj* dan *prep* bertipe *string*, merupakan variabel untuk menyimpan *subject*, *modal*, *frequency*, *verb*, *object* dan *preposition* dari kalimat.

d. Tipe data TEkspresi.

Tipe data TEkspresi merupakan *record* dari *field* kalimat_ke, awal, akhir, dan eks. Berikut ini merupakan deklarasi tipe data TEkspresi :

```
TEkspresi = record
      Kalimat_ke : Integer;
      awal : Integer;
      akhir : Integer;
      eks : String;
end;
```

Field kalimat_ke bertipe *integer*, merupakan variabel untuk menyimpan kalimat ke-n, yaitu kalimat ke berapa yang memiliki ekspresi yang akan disimpan. *Field* awal dan akhir bertipe *integer*, merupakan variabel untuk menyimpan batas awal kalimat dan batas akhir kalimat yang memiliki ekspresi tersebut. *Field* eks bertipe *string*, merupakan variabel untuk menyimpan ekspresi dari kalimat yang bersangkutan.

3.2.1. Contoh Penerapan Tipe Data.

Sesuai dengan desain di atas, misalkan diberikan kalimat "*If John sees movie in the weekend and John doesn't study math, John will get bad score in math*.", maka kalimat ini bila diterapkan pada tipe data TKalimat akan tampak seperti pada Tabel 3.2.

Tabel 3.2. Tabel Penerapan Kalimat Pada Tipe Data TKalimat.

Panjang Kalimat	Kata ke	Teks	Status Proses	Status Teks
20	1	*If*	*FALSE*	I
	2	*John*	*FALSE*	X
	3	*Sees*	*FALSE*	V
	4	*Movie*	*FALSE*	X
	5	*In*	*FALSE*	P
	6	*The*	*FALSE*	X
	7	*Weekend*	*FALSE*	X
	8	*And*	*FALSE*	K
	9	*John*	*FALSE*	X
	10	*Doesn't*	*FALSE*	N
	11	*Study*	*FALSE*	V
	12	*Math*	*FALSE*	X
	13	*,*	*FALSE*	,
	14	*John*	*FALSE*	X
	15	*Will*	*FALSE*	M
	16	*Get*	*FALSE*	V
	17	*Bad*	*FALSE*	X
	18	*Score*	*FALSE*	X
	19	*In*	*FALSE*	P
	20	*Math*	*FALSE*	X

Pada Tabel 3.2 di atas tampak bahwa *field* status_proses kata masih bernilai *false*, karena belum diproses, dan *field* status_teks dari kata “John”, “*movie*”, “*the*”, “*weekend*”, “*math*”, “*bad*”, dan “*score*” juga masih bernilai “X”, karena belum diproses, dan kata tersebut bukan merupakan *keyword*. Sedangkan ketika kalimat sudah diproses maka akan terjadi perubahan pada tipe data TKalimat. Perubahan pada tipe data TKalimat setelah kalimat diproses dapat dilihat pada Tabel 3.3. Tampak pada *field* status_proses telah bernilai True. Dan pada *field* status_teks sudah diketahui mana yang *subject* dan mana yang *object*.

Tabel 3.3. Tabel Penerapan Kalimat Sudah Diproses Pada Tipe Data TKalimat.

Panjang Kalimat	Kata ke	Teks	Status Proses	Status Teks
20	1	*If*	*TRUE*	I
	2	*John*	*TRUE*	S
	3	*Sees*	*TRUE*	V
	4	*Movie*	*TRUE*	O
	5	*In*	*TRUE*	P
	6	*The*	*TRUE*	O
	7	*Weekend*	*TRUE*	O
	8	*And*	*TRUE*	K
	9	*John*	*TRUE*	S
	10	*Doesn't*	*TRUE*	N
	11	*Study*	*TRUE*	V
	12	*Math*	*TRUE*	O
	13	*,*	*TRUE*	,
	14	*John*	*TRUE*	S
	15	*Will*	*TRUE*	M
	16	*Get*	*TRUE*	V
	17	*Bad*	*TRUE*	O
	18	*Score*	*TRUE*	O
	19	*In*	*TRUE*	P
	20	*Math*	*TRUE*	O

Sesuai dengan kalimat yang diberikan di atas, maka apabila kalimat tersebut diproses akan menghasilkan tiga translasi. Ketiga translasi tersebut dapat dilihat pada Tabel 3.4, Tabel 3.5 dan Tabel 3.6.

Tabel 3.4. Tabel Penerapan Kalimat Pada Tipe Data TTranslasi (Satu).

Kalimat Ke	1
Awal	2
Akhir	7
Huruf	a
Ingkaran	*FALSE*
Subject	*John*
Tobe	
Freq	
Verb	*Sees*
Obj	*Movie*
Prep	*In the weekend*

Tabel 3.5. Tabel Penerapan Kalimat Pada Tipe Data TTraslasi (Dua).

Kalimat Ke	1
Awal	9
Akhir	12
Huruf	b
Ingkaran	*TRUE*
Subject	*John*
Tobe	*Does*
Freq	
Verb	*Study*
Obj	*Math*
Prep	

Tabel 3.6. Tabel Penerapan Kalimat Pada Tipe Data TTraslasi (Tiga).

Kalimat Ke	1
Awal	14
Akhir	20
Huruf	c
Ingkaran	*FALSE*
Subject	*John*
Tobe	*Will*
Freq	
Verb	*Get*
Obj	*bad score*
Prep	*in math*

Selain disimpan pada tipe data TTranslasi, kalimat yang sudah diproses juga akan disimpan pada tipe data TEskpresi. Sesuai dengan kalimat yang diberikan di atas, maka akan dihasilkan tiga ekspresi. Ketiga ekspresi tersebut dapat dilihat pada Tabel 3.7, Tabel 3.8 dan Tabel 3.9.

Tabel 3.7. Tabel Penerapan Kalimat Pada Tipe Data TEkspresi (Satu).

Kalimat Ke	1
Awal	9
Akhir	12
Eks	¬b

Tabel 3.8. Tabel Penerapan Kalimat Pada Tipe Data TEkspresi (Dua).

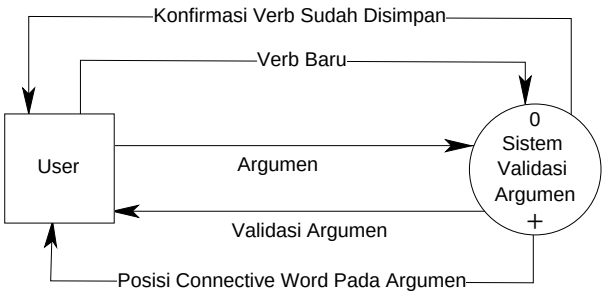
Kalimat Ke	1
Awal	2
Akhir	12
Eks	$a \wedge \neg b$

Tabel 3.9. Tabel Penerapan Kalimat Pada Tipe Data TEkspresi (Tiga).

Kalimat Ke	1
Awal	1
Akhir	20
Eks	$(a \wedge \neg b) \rightarrow c$

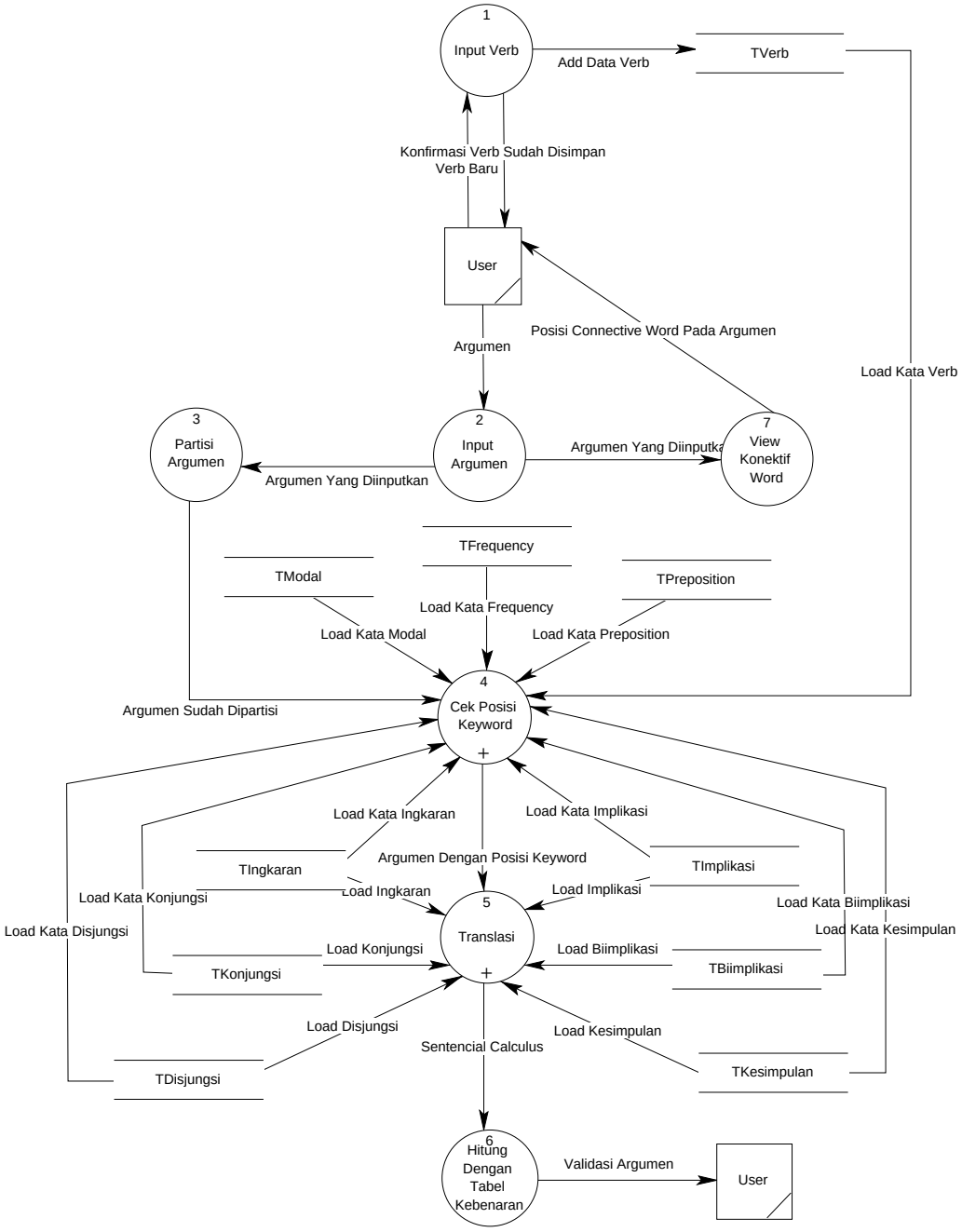
3.3. *Data Flow Diagram*

Data Flow Diagram dari program pengecekan validasi argumen berbahasa Inggris dapat dilihat pada gambar 3.1. Dalam *Data Flow Diagram* program pengecekan validasi argumen berbahasa Inggris hanya menggunakan satu *entity*, yaitu *user* yang menggunakan program pengecekan validasi argumen berbahasa Inggris ini.



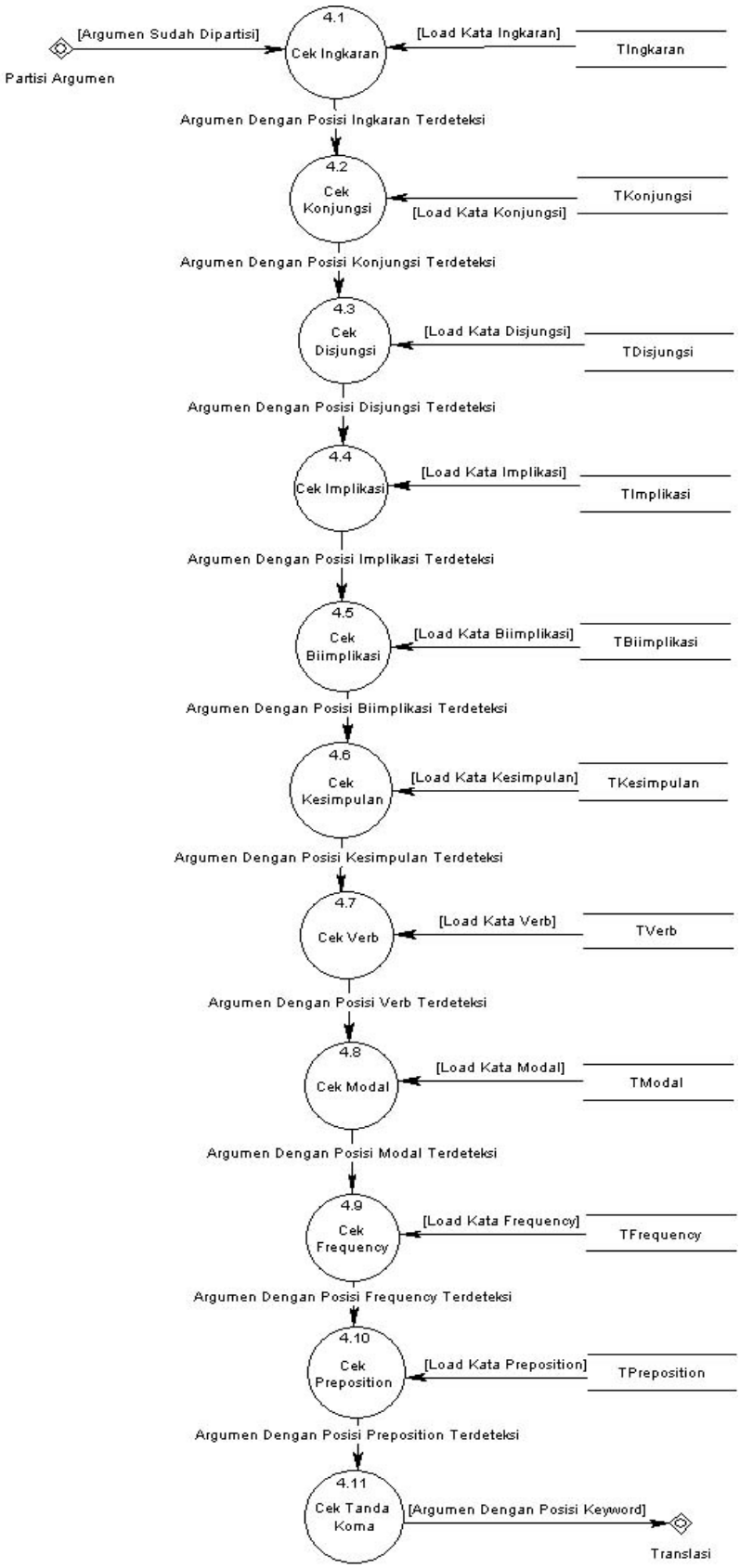
Gambar 3.1. DFD *Context Diagram* Sistem Validasi Argumen Berbahasa Inggris.

Sistem Validasi Argumen Berbahasa Inggris terdiri dari beberapa proses, sehingga *context diagram* di atas dipecah menjadi beberapa proses dalam DFD level 0 seperti pada Gambar 3.2.



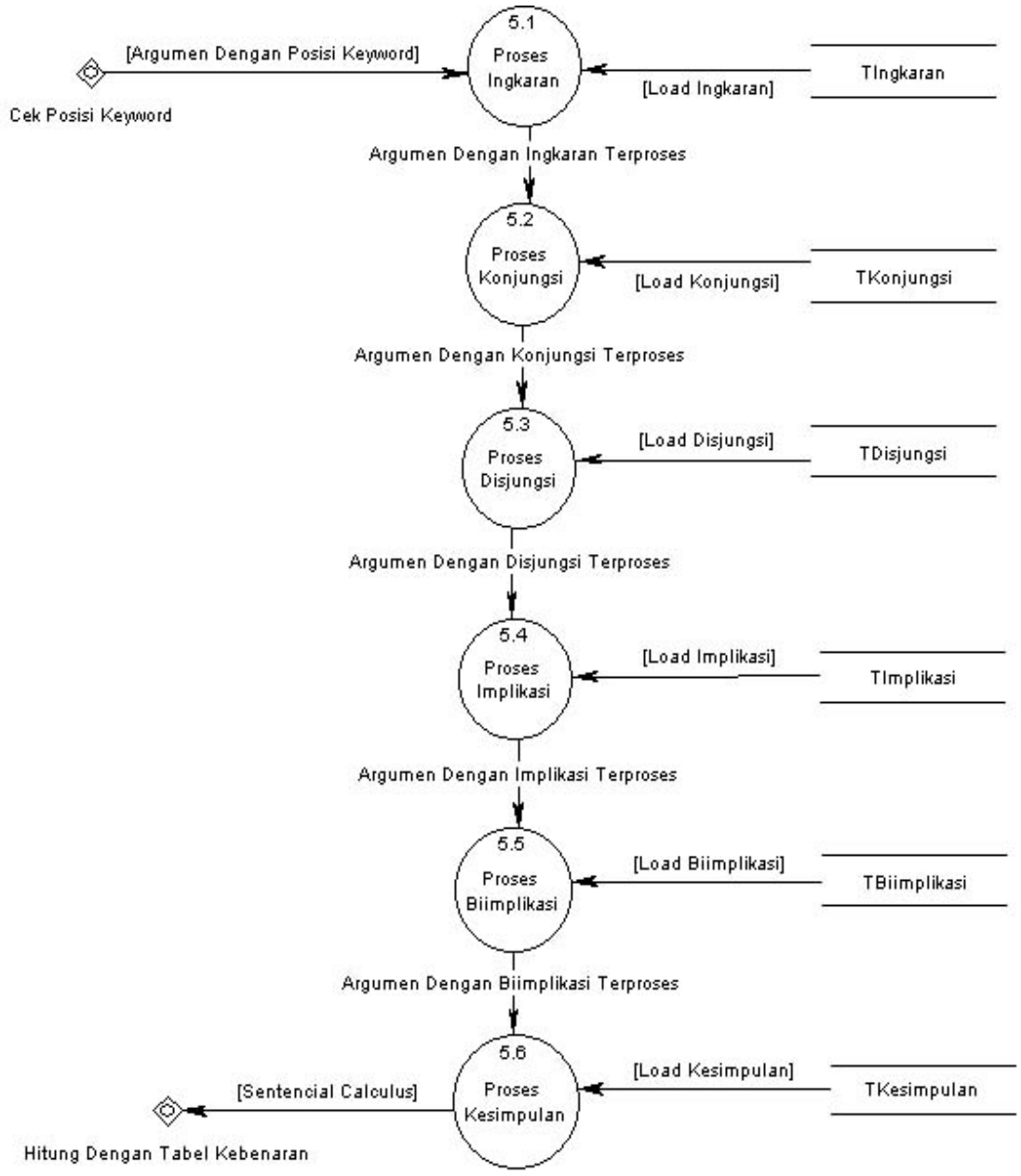
Gambar 3.2. DFD Level 0 Sistem Validasi Argumen Berbahasa Inggris.

Proses 4 dan Proses 5 dalam DFD *level* 0 dipecah lagi menjadi beberapa subproses. Untuk subproses-subproses dari Cek Posisi *Keyword* dapat dilihat pada Gambar 3.3.



Gambar 3.3. DFD Level 1 Proses Cek Posisi *Keyword*.

Berikutnya adalah subproses untuk proses Translasi dapat dilihat pada Gambar 3.4.

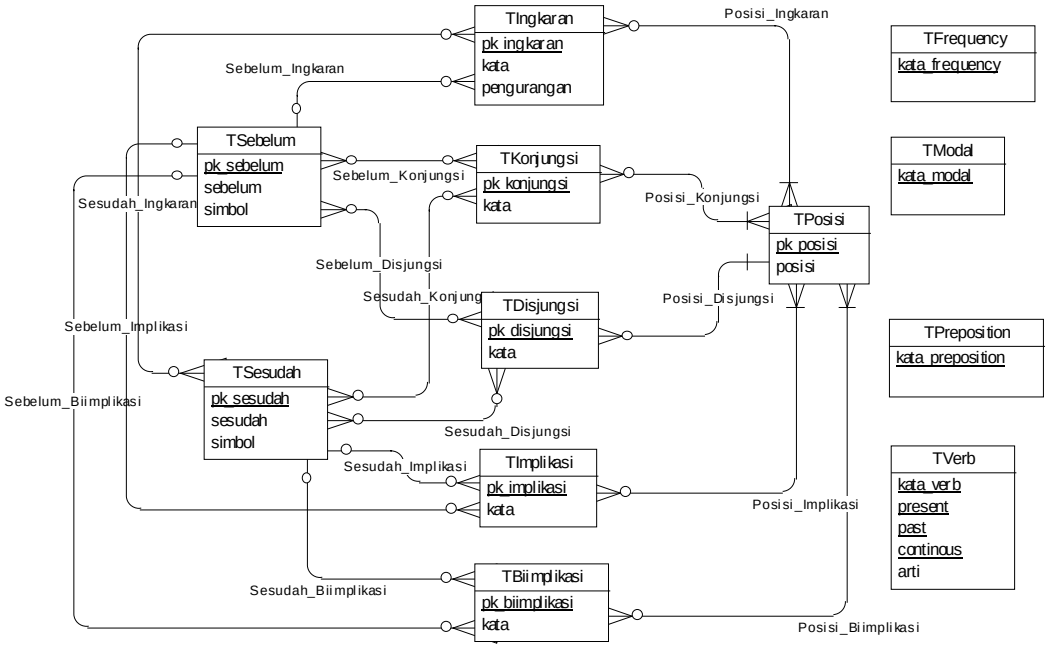


Gambar 3.4. DFD Level 1 Proses Translasi.

3.4. *Entity Relationship Diagram*

Pada proses cek posisi *keyword* dan translasi dibutuhkan pengaksesan database. Oleh karena itu dirancang suatu *database relational* dalam bentuk *Entity*

Relationship Diagram (ERD). ERD *Conceptual Data Model* dari sistem dapat dilihat pada Gambar 3.5.



Gambar 3.5. *Entity Relationship Diagram Conceptual Data Model*.

3.5. Desain Tabel

Sesuai dengan *Entity Relationship Diagram* (ERD) yang telah dirancang di atas, dibentuk suatu *database* yang terdiri dari tabel-tabel berikut :

a. Tabel TIngkaran, yang digunakan untuk menyimpan kata-kata yang termasuk dalam konektif ingkaran. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_ingkaran : untuk menyimpan kode ingkaran sebagai *primary key*.
   - kata : untuk menyimpan kata yang termasuk ingkaran.
   - pk_sebelum : untuk menyimpan kode sebelum yang nantinya digunakan pada Tsebelum untuk melihat kata sebelumnya dari ingkaran yang bersangkutan.
   - pengurangan : untuk menyimpan jumlah pengurangan agar mendapatkan kata yang sesungguhnya.

Tabel 3.10. Tabel TIngkaran.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_ingkaran	*Text*	5
kata	*Text*	20
pk_sebelum	*Text*	5
pengurangan	Number	Integer

b. Tabel TKonjungsi, yang digunakan untuk menyimpan kata-kata yang termasuk dalam konektif konjungsi. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_konjungsi : untuk menyimpan kode konjungsi sebagai *primary key*.
   - kata : untuk menyimpan kata yang termasuk konjungsi.

Tabel 3.11. Tabel TKonjungsi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_konjungsi	*Text*	5
kata	*Text*	20

c. Tabel TDisjungsi, yang digunakan untuk menyimpan kata-kata yang termasuk dalam konektif disjungsi. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_disjungsi : untuk menyimpan kode disjungsi sebagai *primary key*.
   - kata : untuk menyimpan kata yang termasuk disjungsi.
   - pk_posisi : untuk menyimpan kode posisi yang nantinya digunakan pada TPosisi untuk melihat posisi kata disjungsi yang bersangkutan.

Tabel 3.12. Tabel TDisjungsi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_disjungsi	*Text*	5
Kata	*Text*	20
pk_posisi	*Text*	5

d. Tabel TImplikasi, yang digunakan untuk menyimpan kata-kata yang termasuk dalam konektif implikasi. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_implikasi : untuk menyimpan kode implikasi sebagai *primary key*.
   - kata : untuk menyimpan kata yang termasuk implikasi.

- pk_sebelum : untuk menyimpan kode sebelum yang nantinya digunakan pada TSebelum untuk melihat kata sebelumnya dari implikasi yang bersangkutan.
- pk_sesudah : untuk menyimpan kode sesudah yang nantinya digunakan pada TSesudah untuk melihat kata sesudahnya dari implikasi yang bersangkutan.

Tabel 3.13. Tabel TImplikasi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_implikasi	*Text*	5
Kata	*Text*	20
pk_sebelum	*Text*	5
pk_sesudah	*Text*	5

e. Tabel TBiimplikasi, yang digunakan untuk menyimpan kata-kata yang termasuk dalam konektif biimplikasi. Tabel ini memiliki *field-field* sebagai berikut:

- pk_biimplikasi : untuk menyimpan kode biimplikasi sebagai *primary key*.
- kata : untuk menyimpan kata yang termasuk biimplikasi.
- pk_sebelum : untuk menyimpan kode sebelum yang nantinya digunakan pada TSebelum untuk melihat kata sebelumnya dari biimplikasi yang bersangkutan.
- pk_sesudah : untuk menyimpan kode sesudah yang nantinya digunakan pada TSesudah untuk melihat kata sesudahnya dari biimplikasi yang bersangkutan.

Tabel 3.14. Tabel TBiimplikasi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_biimplikasi	*Text*	5
Kata	*Text*	20
pk_sebelum	*Text*	5
pk_sesudah	*Text*	5

f. Tabel TKesimpulan, yang digunakan untuk menyimpan kata-kata yang termasuk dalam kesimpulan. Tabel ini memiliki *field-field* sebagai berikut:

- pk_kesimpulan : untuk menyimpan kode kesimpulan sebagai *primary key*.
- kata : untuk menyimpan kata yang termasuk kesimpulan.

Tabel 3.15. Tabel TKesimpulan.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_kesimpulan	*Text*	5
Kata	*Text*	20

g. Tabel TPosisi, yang digunakan untuk menyimpan posisi dari kata. Tabel ini memiliki *field-field* sebagai berikut:

- pk_posisi : untuk menyimpan kode posisi sebagai *primary key*.
- posisi : untuk menyimpan posisi

Tabel 3.16. Tabel TPosisi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_posisi	*Text*	5
Posisi	*Text*	10

h. Tabel TMMPosisiIngkaran, yang digunakan untuk menyimpan posisi kata-kata ingkaran. Tabel ini memiliki *field-field* sebagai berikut:

- pk_ingkaran : untuk menyimpan kode ingkaran sebagai *primary key*.
- pk_posisi : untuk menyimpan kode posisi sebagai *primary key*.

Tabel 3.17. Tabel TMMPosisi Ingkaran.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_ingkaran	*Text*	5
pk_posisi	*Text*	5

i. Tabel TMMPosisiKonjungsi, yang digunakan untuk menyimpan posisi kata-kata konjungsi. Tabel ini memiliki *field-field* sebagai berikut:

- pk_konjungsi : untuk menyimpan kode konjungsi sebagai *primary key*.
- pk_posisi : untuk menyimpan kode posisi sebagai *primary key*.

Tabel 3.18. Tabel TMMPosisi Konjungsi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_konjungsi	*Text*	5
pk_posisi	*Text*	5

j. Tabel TMMPosisiImplikasi, yang digunakan untuk menyimpan posisi kata-kata implikasi. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_implikasi : untuk menyimpan kode implikasi sebagai *primary key*.
   - pk_posisi : untuk menyimpan kode posisi sebagai *primary key*.

Tabel 3.19. Tabel TMMPosisi Implikasi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_implikasi	*Text*	5
pk_posisi	*Text*	5

k. Tabel TMMPosisiBiimplikasi, yang digunakan untuk menyimpan posisi kata-kata biimplikasi. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_biimplikasi : untuk menyimpan kode biimplikasi sebagai *primary key*.
   - pk_posisi : untuk menyimpan kode posisi sebagai *primary key*.

Tabel 3.20. Tabel TMMPosisi Biimplikasi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_biimplikasi	*Text*	5
pk_posisi	*Text*	5

l. Tabel TSebelum, yang digunakan untuk menyimpan kata-kata yang letaknya memungkinkan sebelum *keyword*. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_sebelum : untuk menyimpan kode kata-kata yang letaknya memungkinkan sebelum *keyword*, dan kode tersebut fungsinya sebagai *primary key*.
   - kata : untuk menyimpan kata yang letaknya memungkinkan sebelum *keyword*.

- simbol : untuk menyimpan simbol dari kata yang letaknya memungkinkan sebelum *keyword*.

Tabel 3.21. Tabel TSebelum.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_sebelum	*Text*	5
kata	*Text*	20
simbol	*Text*	1

m. Tabel TSesudah, yang digunakan untuk menyimpan kata-kata yang letaknya memungkinkan sesudah *keyword*. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_sesudah : untuk menyimpan kode kata-kata yang letaknya memungkinkan sesudah *keyword*, dan kode tersebut fungsinya sebagai *primary key*.
   - kata : untuk menyimpan kata yang letaknya memungkinkan sesudah *keyword*.
   - Simbol : untuk menyimpan simbol dari kata yang letaknya memungkinkan sesudah *keyword*.

Tabel 3.22. Tabel TSesudah.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_sesudah	*Text*	5
kata	*Text*	20
simbol	*Text*	1

n. Tabel TSebelumKonjungsi, yang digunakan untuk menyimpan kata-kata sebelum konjungsi. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_konjungsi : untuk menyimpan kode konjungsi sebagai *primary key*.
   - pk_sebelum : untuk menyimpan kode sebelum sebagai *primary key*.

Tabel 3.23. Tabel TSebelumKonjungsi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_konjungsi	*Text*	5
pk_sebelum	*Text*	5

o. Tabel TSebelumDisjungsi, yang digunakan untuk menyimpan kata-kata sebelum disjungsi. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_disjungsi : untuk menyimpan kode disjungsi sebagai *primary key*.
   - pk_sebelum : untuk menyimpan kode sebelum sebagai *primary key*.

Tabel 3.24. Tabel TSebelumDisjungsi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_disjungsi	*Text*	5
pk_sebelum	*Text*	5

p. Tabel TSesudahIngkaran, yang digunakan untuk menyimpan kata-kata sesudah ingkaran. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_ingkaran : untuk menyimpan kode ingkaran sebagai *primary key*.
   - pk_sesudah : untuk menyimpan kode sesudah sebagai *primary key*.

Tabel 3.25. Tabel TSesudahIngkaran.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_ingkaran	*Text*	5
pk_sesudah	*Text*	5

q. Tabel TSesudahKonjungsi, yang digunakan untuk menyimpan kata-kata sesudah konjungsi. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_konjungsi : untuk menyimpan kode konjungsi sebagai *primary key*.
   - pk_sesudah : untuk menyimpan kode sesudah sebagai *primary key*.

Tabel 3.26. Tabel TSesudahKonjungsi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_konjungsi	*Text*	5
pk_sesudah	*Text*	5

r. Tabel TSesudahDisjungsi, yang digunakan untuk menyimpan kata-kata sesudah disjungsi. Tabel ini memiliki *field-field* sebagai berikut:
   - pk_disjungsi : untuk menyimpan kode disjungsi sebagai *primary key*.
   - pk_sesudah : untuk menyimpan kode sesudah sebagai *primary key*.

Tabel 3.27. Tabel TSesudahDisjungsi.

Nama *Field*	Tipe *Field*	Ukuran *Field*
pk_disjungsi	*Text*	5
pk_sesudah	*Text*	5

s. Tabel T*Modal*, yang digunakan untuk menyimpan kata-kata yang merupakan *modal* dalam bahasa Inggris. Tabel ini memiliki *field-field* sebagai berikut:
   - kata_*modal* : untuk menyimpan kata-kata yang merupakan *modal* dalam bahasa Inggris dan *field* ini sekaligus merupakan *primary key*.

Tabel 3.28. Tabel T*Modal*.

Nama *Field*	Tipe *Field*	Ukuran *Field*
kata_*modal*	*Text*	20

t. Tabel T*Frequency*, yang digunakan untuk menyimpan kata-kata yang merupakan *frequency* dalam bahasa Inggris. Tabel ini memiliki *field-field* sebagai berikut:
   - kata_*frequency* : untuk menyimpan kata-kata yang merupakan *frequency* dalam bahasa Inggris dan *field* ini sekaligus merupakan *primary key*.

Tabel 3.29. Tabel T*Frequency*.

Nama *Field*	Tipe *Field*	Ukuran *Field*
kata_*frequency*	*Text*	20

u. Tabel T*Verb*, yang digunakan untuk menyimpan kata-kata yang merupakan *verb* dalam bahasa Inggris. Tabel ini memiliki *field-field* sebagai berikut:
   - kata_*verb* : untuk menyimpan kata-kata yang merupakan *verb* dasar dalam bahasa Inggris dan *field* ini sekaligus merupakan *primary key*.
   - *present* : untuk menyimpan bentuk *present tense* dari *verb*.
   - *past* : untuk menyimpan bentuk *past* dari *verb*.
   - *continous* : untuk menyimpan bentuk *present continous* dari *verb*.
   - arti : untuk menyimpan terjemahan *verb* ke dalam bahasa Indonesia

Tabel 3.30. Tabel T*Verb*.

Nama *Field*	Tipe *Field*	Ukuran *Field*
kata_*verb*	*Text*	30
present	*Text*	30
past	*Text*	30
continous	*Text*	30
arti	*Text*	255

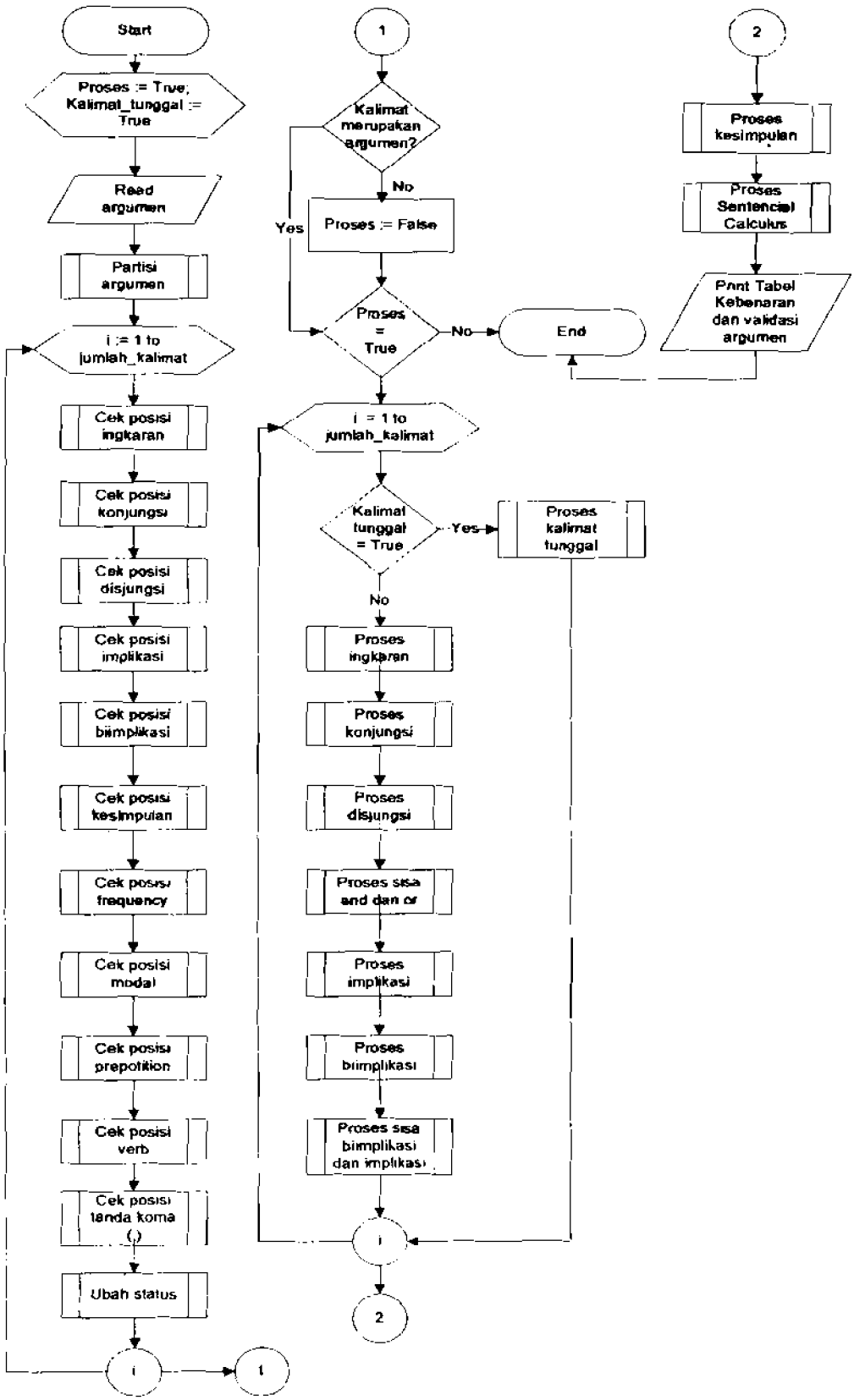
v. Tabel T*Preposition*, yang digunakan untuk menyimpan kata-kata yang merupakan *preposition* dalam bahasa Inggris. Tabel ini memiliki *field-field* sebagai berikut:
    - kata_*preposition* : untuk menyimpan kata-kata yang merupakan *preposition* dalam bahasa Inggris dan *field* ini sekaligus merupakan *primary key*.

Tabel 3.31. Tabel T*Preposition*.

Nama *Field*	Tipe *Field*	Ukuran *Field*
kata_*preposition*	*Text*	20

3.6. *Flowchart* Sistem

Pada Gambar 3.6 dapat dilihat *flowchart* dari program pengecekan validasi argumen berbahasa Inggris. Sebagian besar proses yang dilakukan dalam *flowchart* tersebut adalah pemanggilan *procedure* dan *function*. *Flowchart* pada Gambar 3.6 menggambarkan proses secara keseluruhan dan merupakan proses utama yang dijalankan pada program pengecekan validasi argumen berbahasa Inggris.



Gambar 3.6. Flowchart Sistem Program Pengecekan Validasi Argumen Berbahasa Inggris.

3.6.1. Penjelasan *Flowchart* Sistem.

Berikut ini akan dijelaskan mengenai *flowchart* sistem program pengecekan validasi argumen berbahasa Inggris, seperti yang terlihat pada Gambar 3.6 di halaman sebelumnya :

a. Pada tahap pertama, akan dilakukan proses menginisialisai variabel yang digunakan dalam program.
b. Pada tahap *read* argumen dilakukan pembacaan argumen yang di-*input*-kan. Baik peng-*input*-an argumen dari *keyboard* maupun dari membuka *file* argumen yang telah disimpan. *Input* yang diberikan berupa argumen berbahasa Inggris.
c. Selanjutnya dilakukan pemanggilan procedure untuk memecahkan kalimat argumen ke dalam bentuk kata per kata.
d. Kemudian dilakukan *looping* dari satu sampai sebanyak jumlah kalimat yang ada dalam argumen, dan dalam proses *looping* tersebut dilakukan beberapa perintah.
e. Beberapa perintah yang dilakukan dalam proses *looping* adalah perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan ingkaran dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan konjungsi dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan disjungsi dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan implikasi dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan biimplikasi dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan kesimpulan dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan *frequency* dalam bahasa Inggris yang terdapat dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan *modal* dalam bahasa Inggris yang terdapat dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan *preposition* dalam bahasa Inggris yang terdapat dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi kata yang merupakan *verb* dalam bahasa Inggris yang

terdapat dalam kalimat, perintah pemanggilan *function* untuk mengecek posisi *tanda koma* yang terdapat dalam kalimat.

f. Kemudian dilakukan pamanggilan *procedure* untuk mengubah status tiap kata yang sudah ditemukan sesuai dengan jenis katanya (ingkaran, konjungsi, dll).
g. Langkah selanjutnya setelah *looping* selesai dilakukan pengecekan apakah yang di-*input*-kan adalah argumen atau bukan. Kalau yang di-*input*-kan bukan argumen maka variabel proses akan diisi dengan nilai *false*. Kemudian dilakukan pengecekan lagi pada variabel proses. Jika bernilai *false* maka program akan berhenti. Jika bernilai *true* maka program akan dilanjutkan ke langkah yang berikutnya.
h. Bila proses bernilai *true*, maka langkah selanjutnya adalah melakukan *looping* dari 1 sampai sebanyak jumlah kalimat yang terdapat dalam argumen. Dan dalam proses *looping* tersebut dilakukan beberapa perintah.
i. Proses pertama setelah memasuki *looping* adalah dilakukan pengecekan apakah kalimat yang akan diproses merupakan kalimat tunggal atau bukan. Kalau merupakan kalimat tunggal akan dilakukan pemanggilan *procedure* untuk memproses kalimat tunggal. Sedangkan jika bukan kalimat tunggal akan dilakukan beberapa perintah.
j. Beberapa perintah yang dilakukan apabila kalimat yang diproses bukan kalimat tunggal adalah perintah pemanggilan *procedure* untuk memproses ingkaran, perintah pemanggilan *procedure* untuk memproses konjungsi, perintah pemanggilan *procedure* untuk memproses disjungsi, perintah pemanggilan *procedure* untuk memproses sisa konjungsi dan disjungsi yang belum diproses, perintah pemanggilan *procedure* untuk memproses implikasi, perintah pemanggilan *procedure* untuk memproses biimplikasi, perintah pemanggilan *procedure* untuk memproses sisa implikasi dan biimplikasi yang belum diproses.
k. Setelah *looping* selesai dilakukan pemanggilan *procedure* untuk memproses kesimpulan.
l. Dari keseluruhan proses akan dihasilkan *sentencial calculus*, kemudian dilakukan pemanggilan *procedure* untuk memproses *sentencial calculus*. Di

dalam proses ini dilakukan perhitungan dengan menggunakan tabel kebenaran dalam logika matematika.

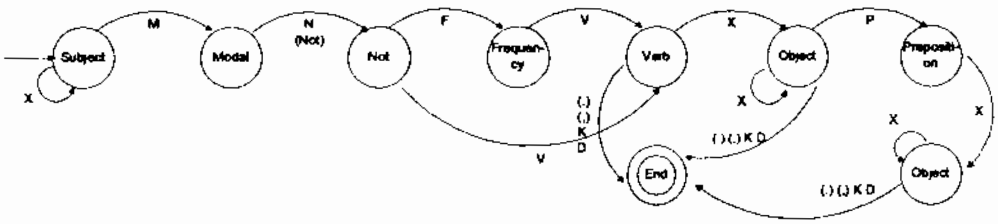
m. Setelah keseluruhan proses selesai akan dihasilkan *output* berupa tabel kebenaran dan hasil kevalidan argumen.

3.7. Desain *Syntax*

Bahasa merupakan sesuatu yang luas dan kompleks. Oleh karena itu diperlukan pembatasan *syntax* pada program. Berikut ini merupakan desain *syntax* yang dapat dikerjakan oleh program pengecekan validasi argumen berbahasa Inggris dalam Tugas Akhir ini :

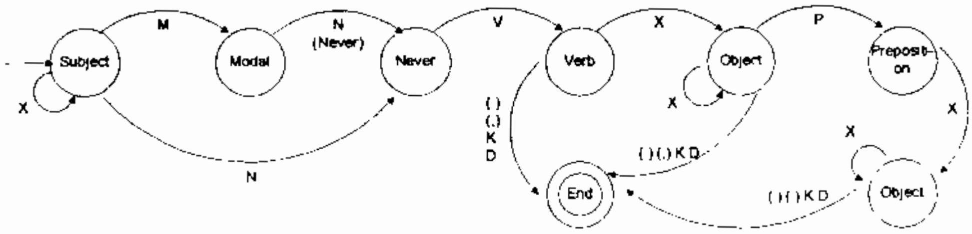
3.7.1. Konektif Ingkaran.

a. *Syntax* kata "*Not*".



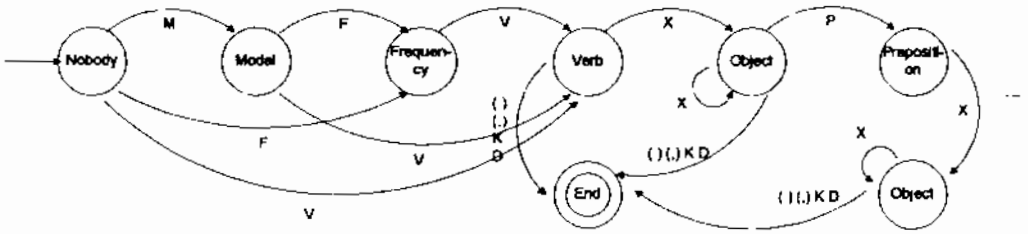
Gambar 3.7. *Syntax* kata "*Not*".

b. *Syntax* kata "*Never*".

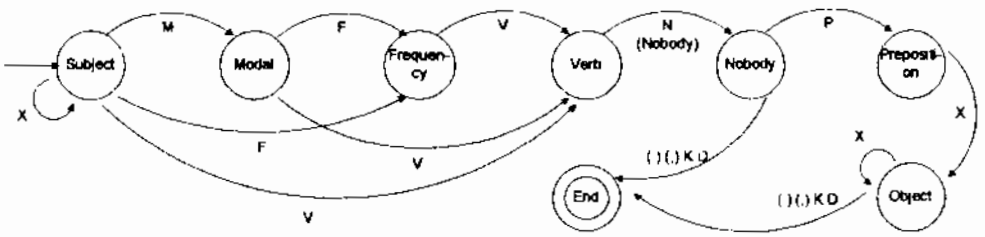


Gambar 3.8. *Syntax* kata "*Never*".

c. *Syntax* kata "*Nobody*".

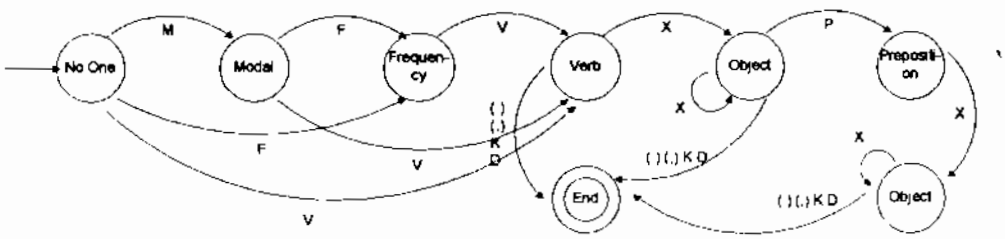


Gambar 3.9. *Syntax* 1 kata "*Nobody*".

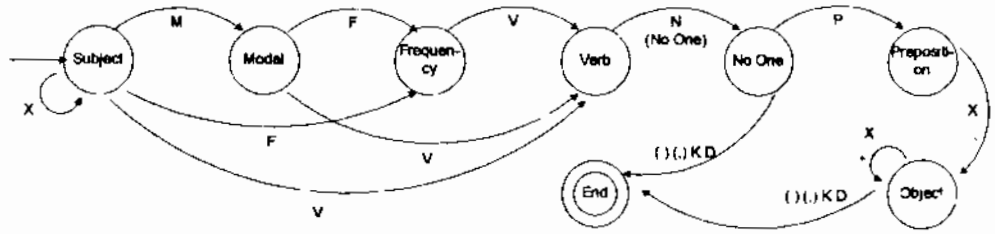


Gambar 3.10. *Syntax* 2 kata "*Nobody*".

d. *Syntax* kata "*No One*".

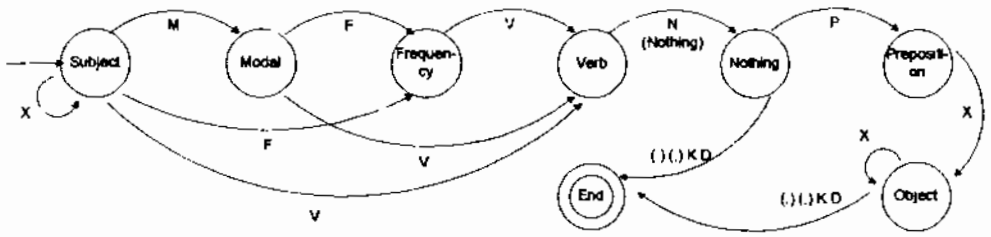


Gambar 3.11. *Syntax* 1 kata "*No One*".



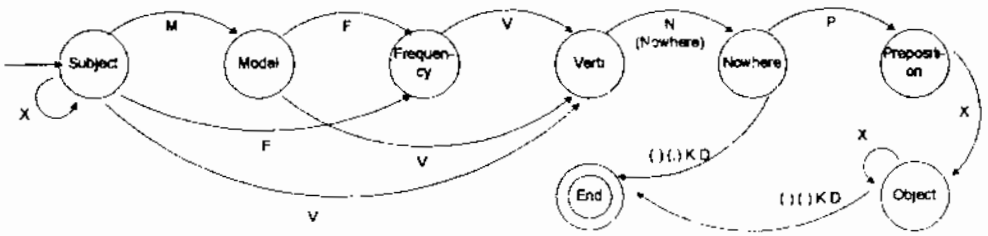
Gambar 3.12. *Syntax* 2 kata "*No One*".

e. *Syntax* kata "*Nothing*".



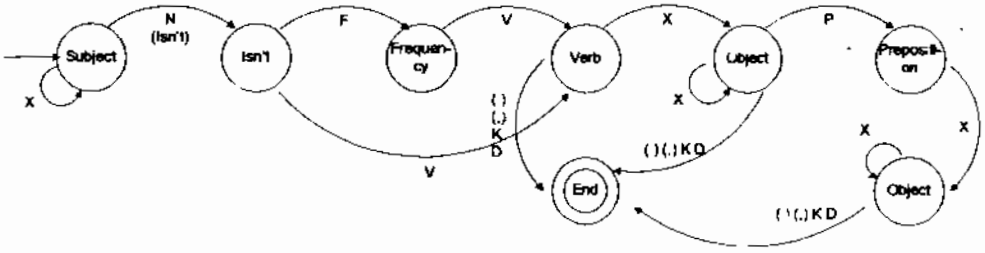
Gambar 3.13. *Syntax* kata "*Nothing*".

f. *Syntax* kata "*Nowhere*".



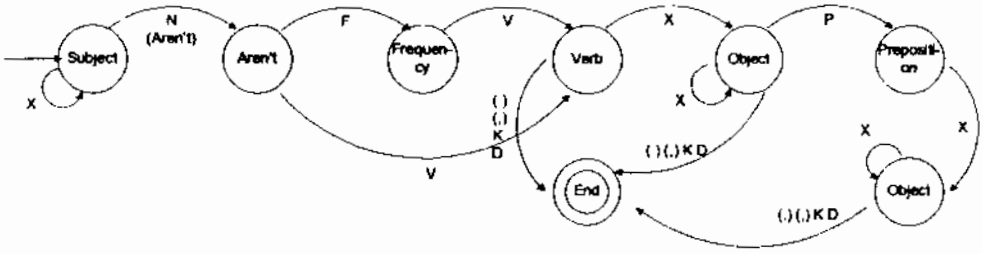
Gambar 3.14. *Syntax* kata "*Nowhere*".

g. *Syntax* kata "*Isn't*".



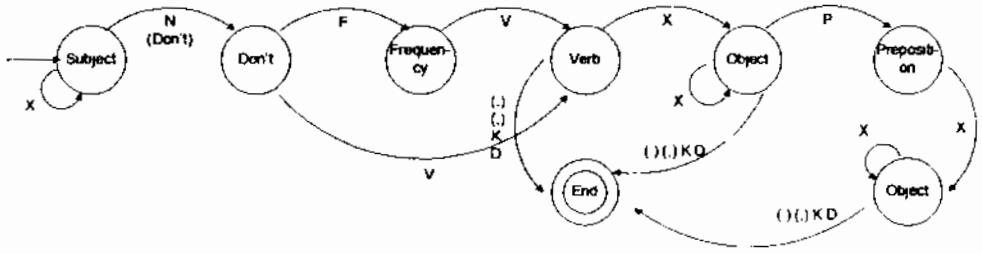
Gambar 3.15. *Syntax* kata "*Isn't*".

h. *Syntax* kata "*Aren't*".



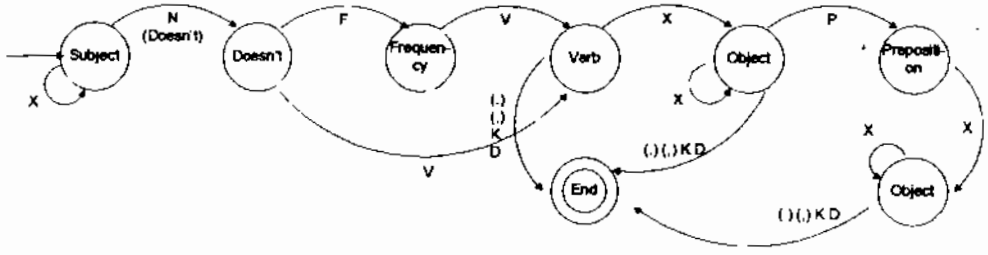
Gambar 3.16. *Syntax* kata "*Aren't*".

i. *Syntax* kata "*Don't*".



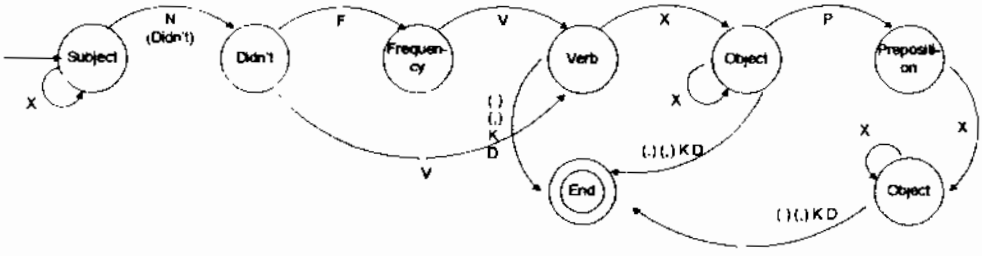
Gambar 3.17. *Syntax* kata "*Don't*".

j. *Syntax* kata "*Doesn't*".



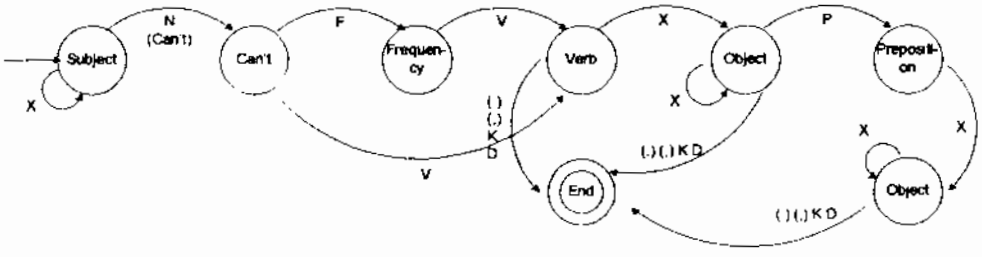
Gambar 3.18. *Syntax* kata "*Doesn't*".

k. *Syntax* kata "*Didn't*".



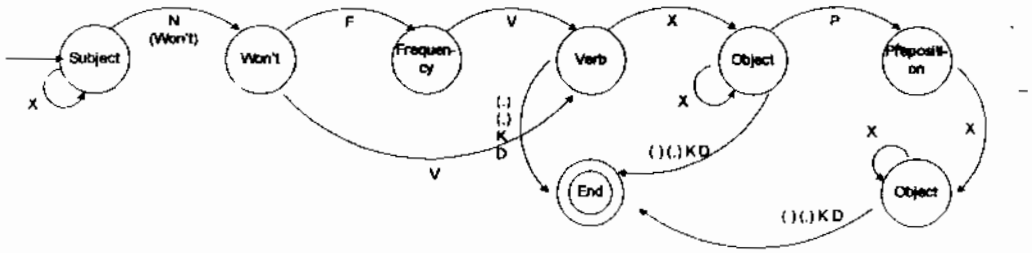
Gambar 3.19. *Syntax* kata "*Didn't*".

l. *Syntax* kata "*Can't*".



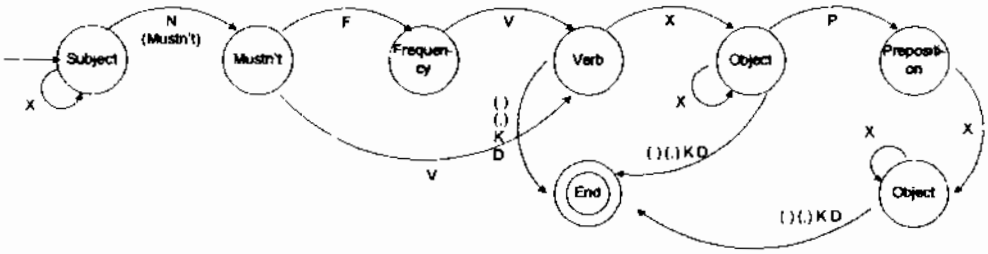
Gambar 3.20. *Syntax* kata "*Can't*".

m. *Syntax* kata "*Won't*".



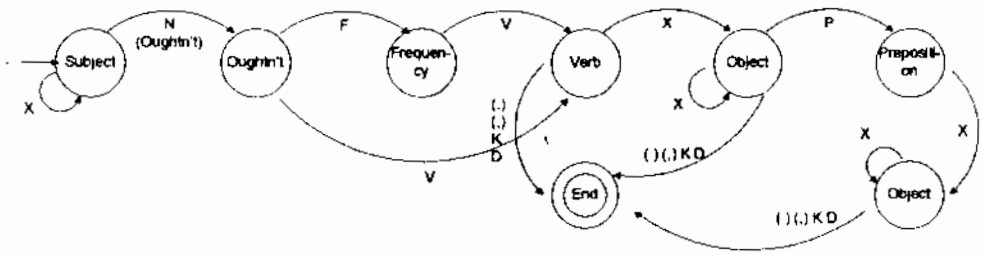
Gambar 3.21. *Syntax* kata "*Won't*".

n. *Syntax* kata "*Mustn't*".



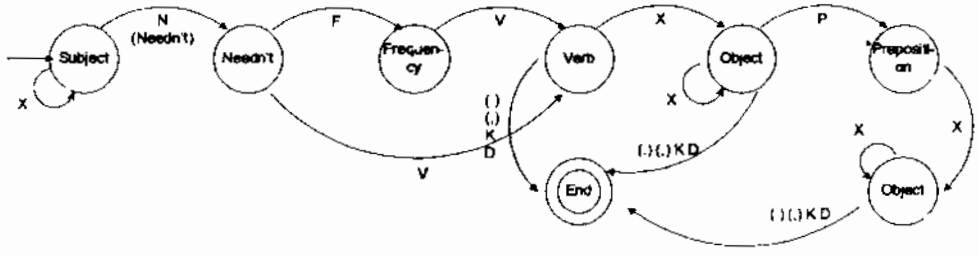
Gambar 3.22. *Syntax* kata "*Mustn't*".

o. *Syntax* kata "*Oughtn't*".



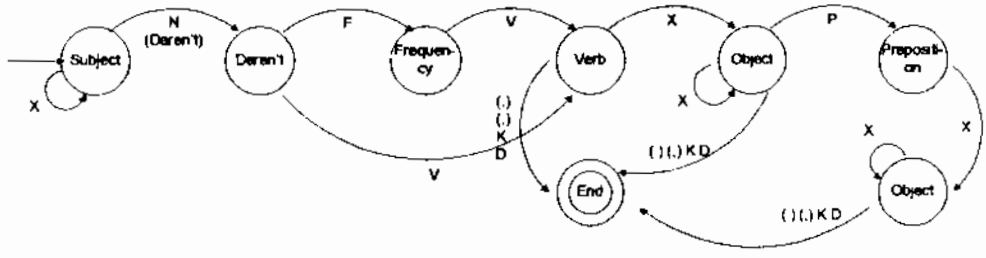
Gambar 3.23. *Syntax* kata "*Oughtn't*".

p. *Syntax* kata "*Needn't*".



Gambar 3.24. *Syntax* kata "*Needn't*".

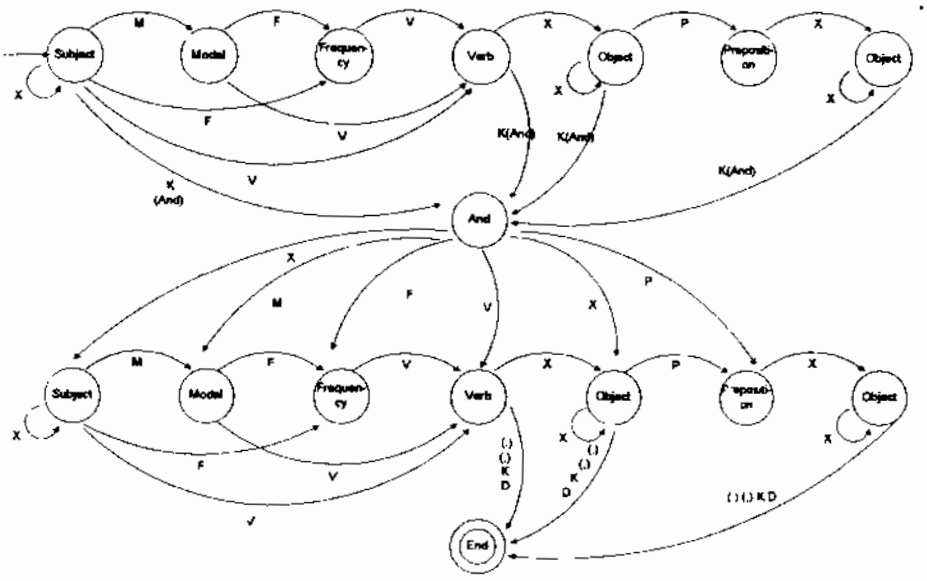
q. *Syntax* kata "*Daren't*".



Gambar 3.25. *Syntax* kata "*Daren't*".

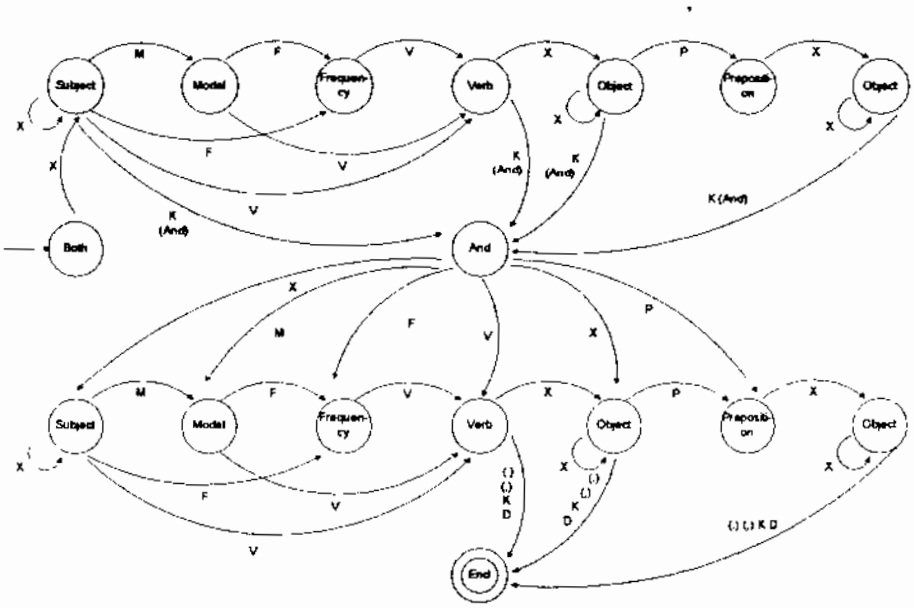
3.7.2. Konektif Konjungsi.

a. *Syntax* kata "*And*".



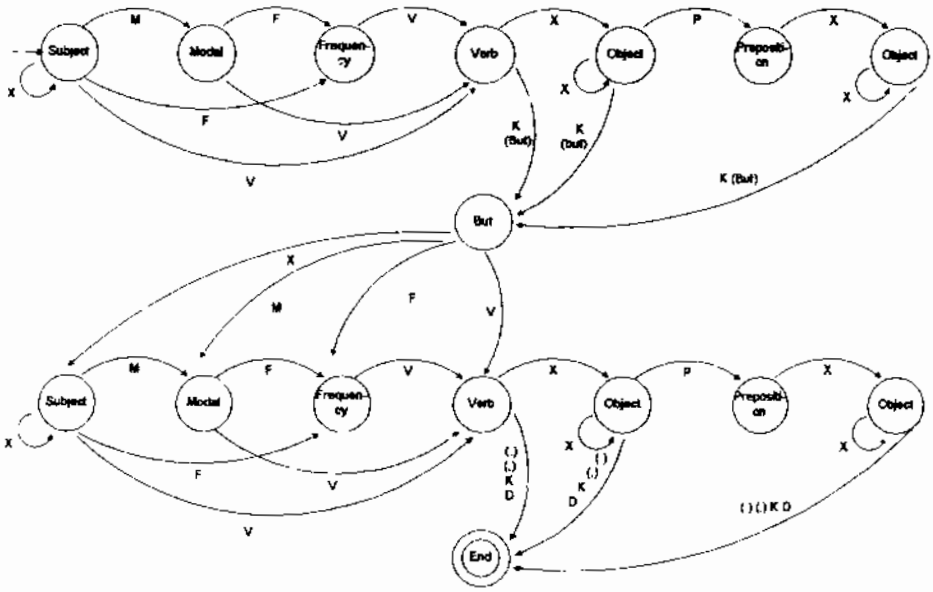
Gambar 3.26. *Syntax* kata "*And*".

b. *Syntax* kata "*Both..And..*".



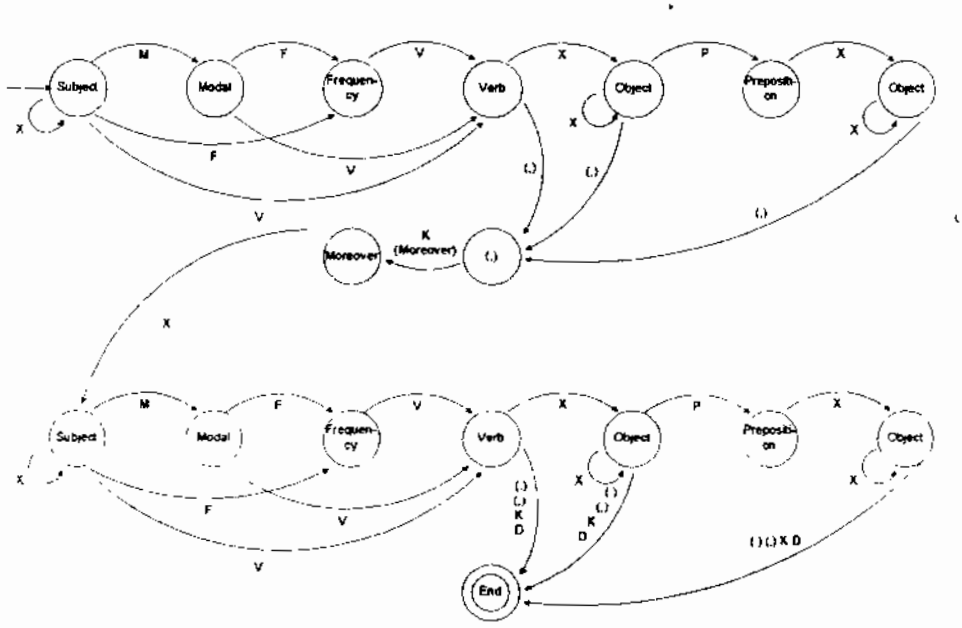
Gambar 3.27. *Syntax* kata "*Both..And..*".

c. *Syntax* kata "*But*".



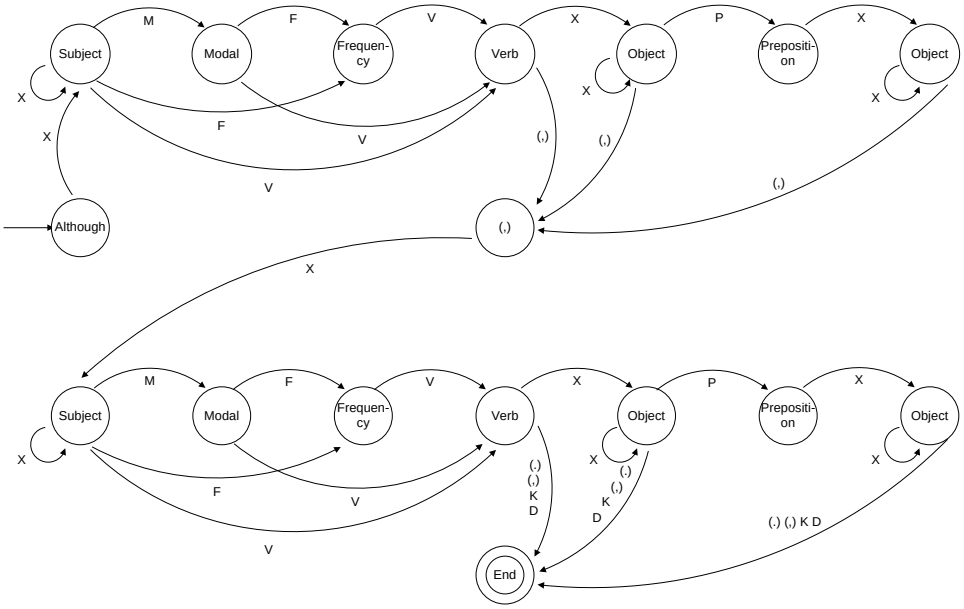
Gambar 3.28. *Syntax* kata "*But*".

d. *Syntax* kata "*Moreover*".

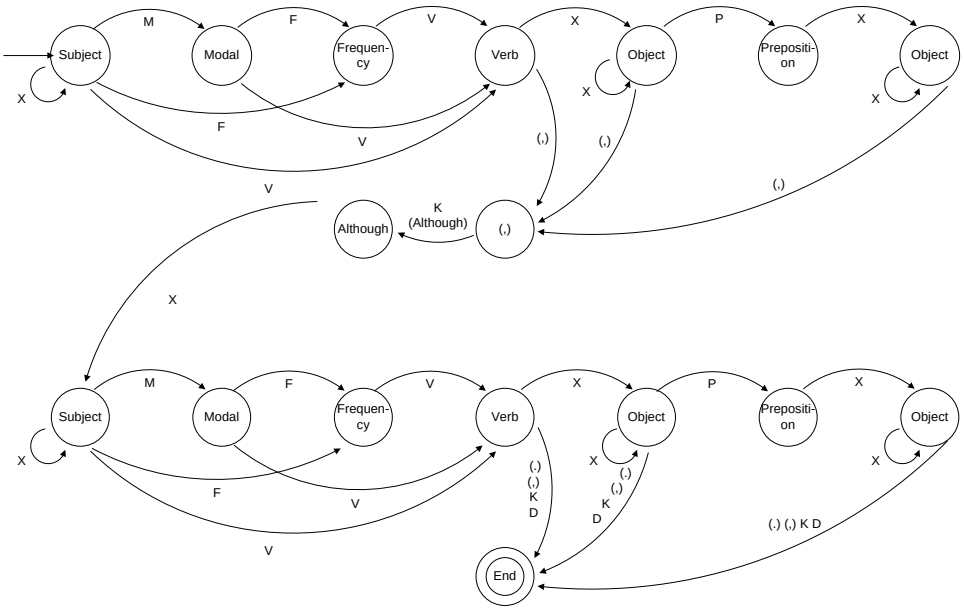


Gambar 3.29. *Syntax* kata "*Moreover*".

e. *Syntax* kata “*Although*”.

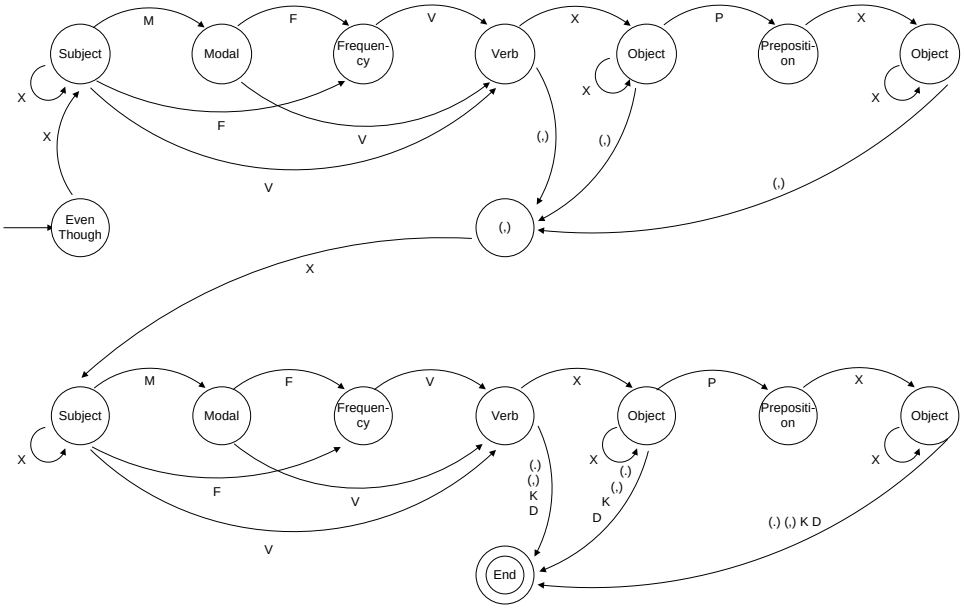


Gambar 3.30. *Syntax* 1 kata “*Although*”.

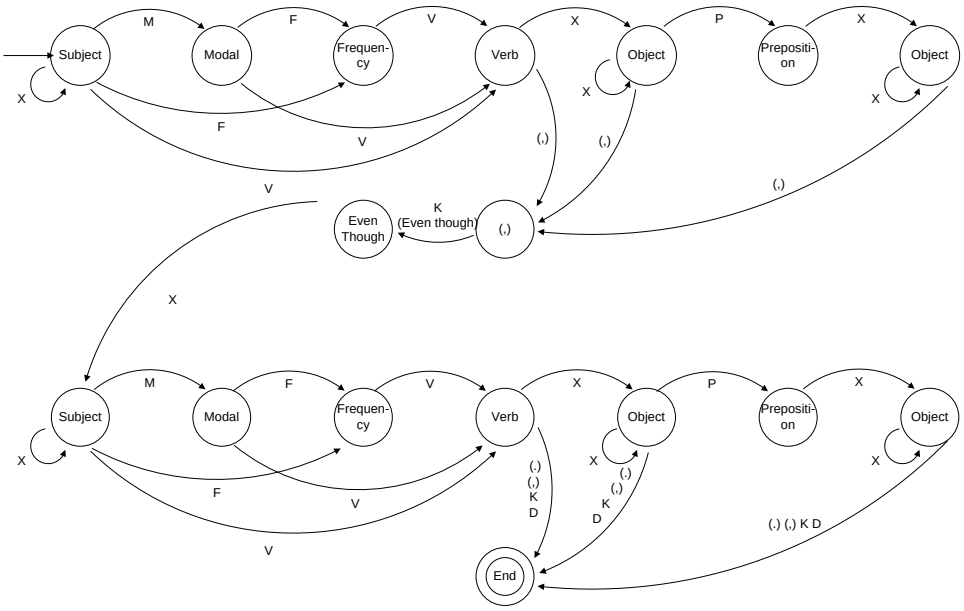


Gambar 3.31. *Syntax* 2 kata “*Although*”.

f. *Syntax* kata "*Even though*"



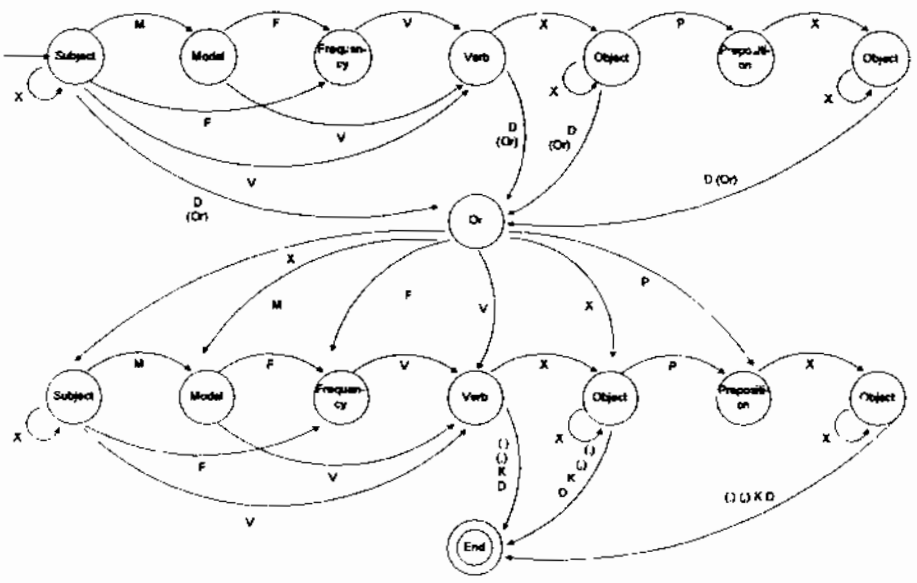
Gambar 3.32. *Syntax* 1 kata "*Even though*".



Gambar 3.33. *Syntax* 2 kata "*Even though*".

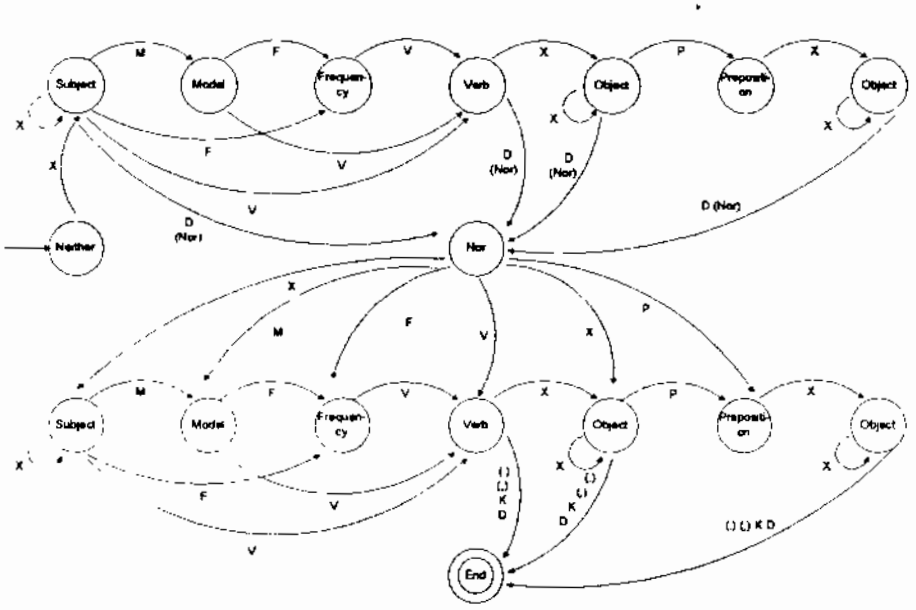
3.7.3. Konektif Disjungsi.

a. *Syntax* kata "*Or*".



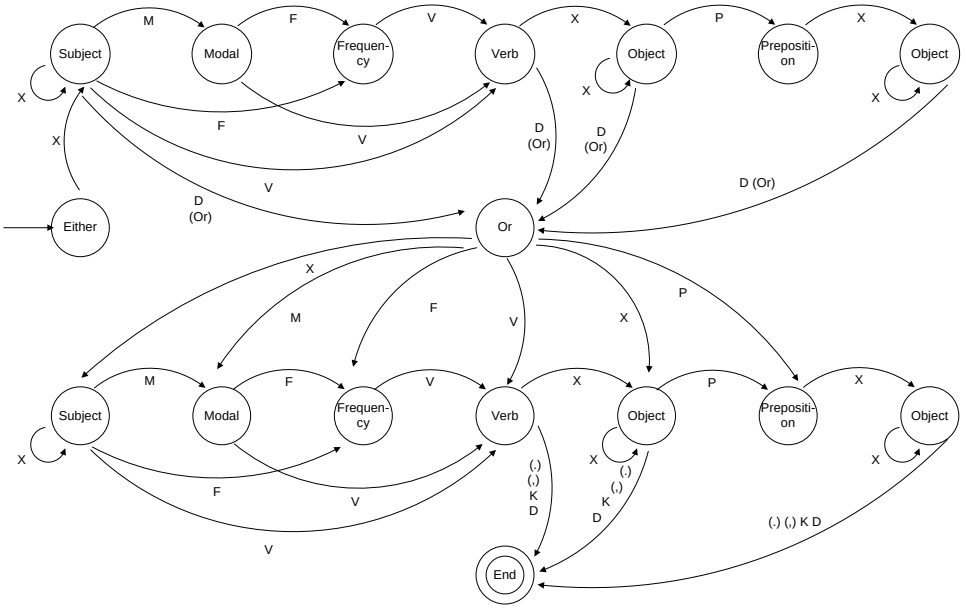
Gambar 3.34. *Syntax* kata "*Or*".

b. *Syntax* kata "*Neither..Nor..*".



Gambar 3.35. *Syntax* kata "*Neither..Nor..*".

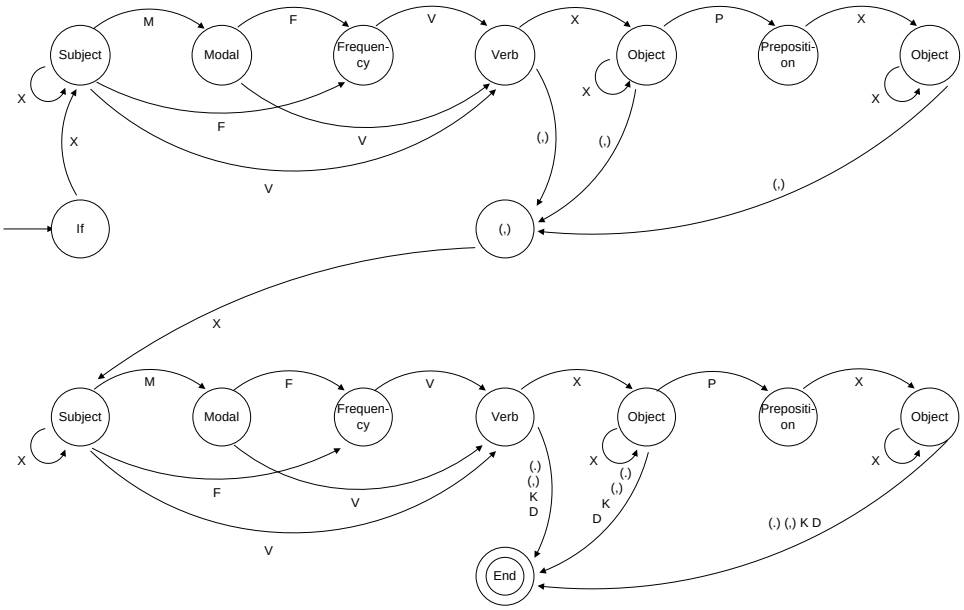
c. *Syntax* kata "*Either..Or..*".



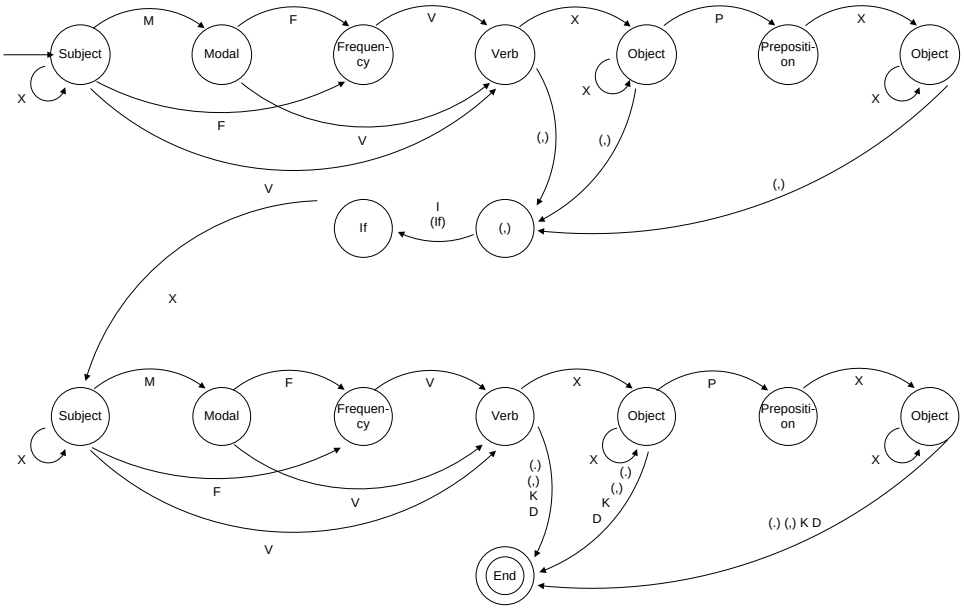
Gambar 3.36. *Syntax* kata "*Either..Or..*".

3.7.4. Konektif Implikasi.

a. *Syntax* kata "*If*".

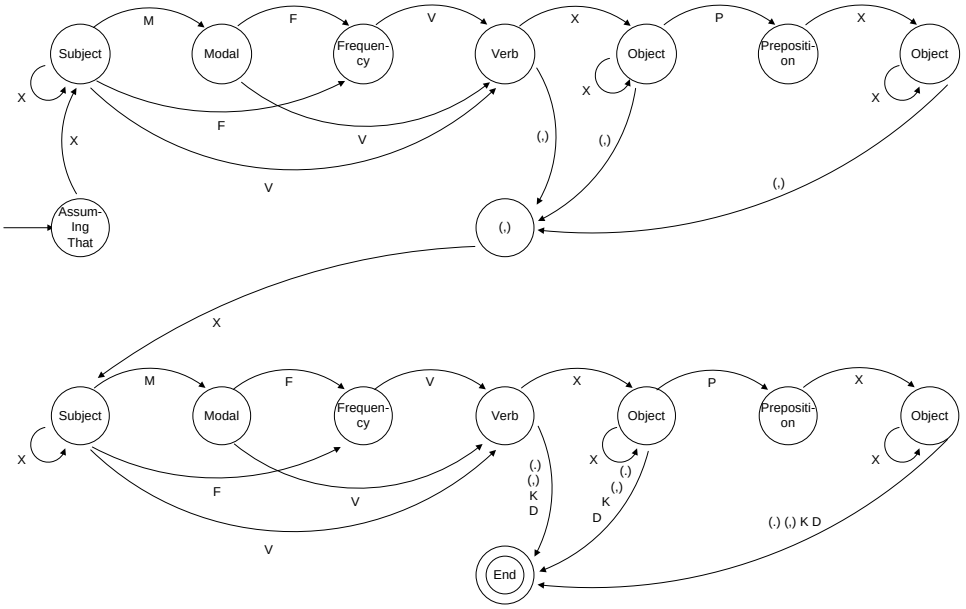


Gambar 3.37. *Syntax* 1 kata "*If*".

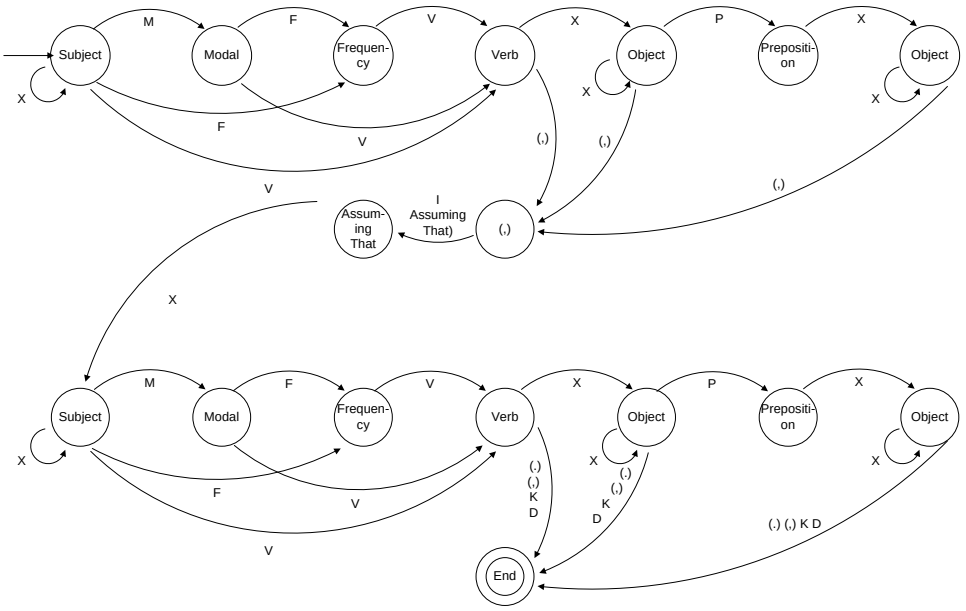


Gambar 3.38. *Syntax* 2 kata "*If*".

b. *Syntax* kata "*Assuming that*".

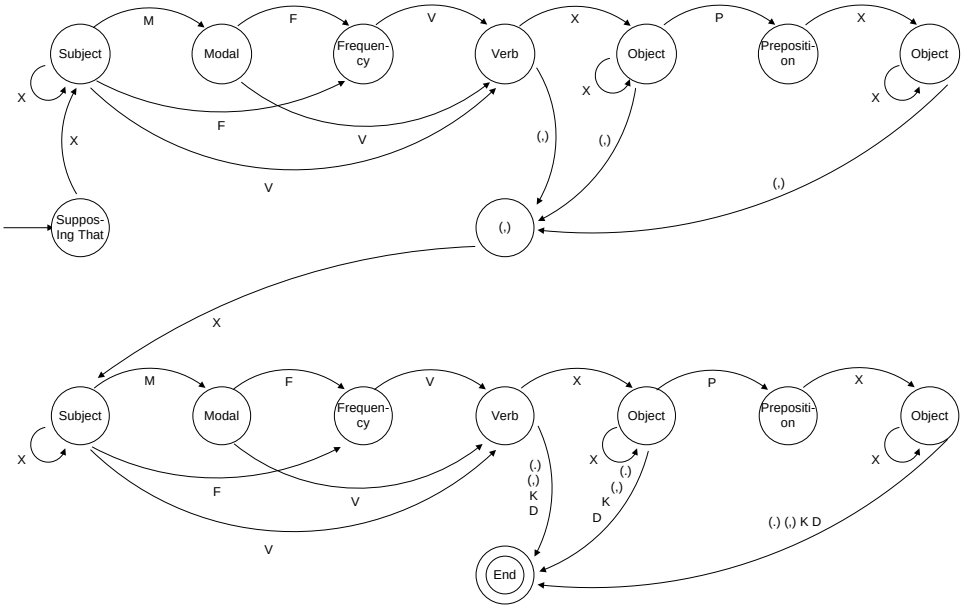


Gambar 3.39. *Syntax* 1 kata "*Assuming that*".

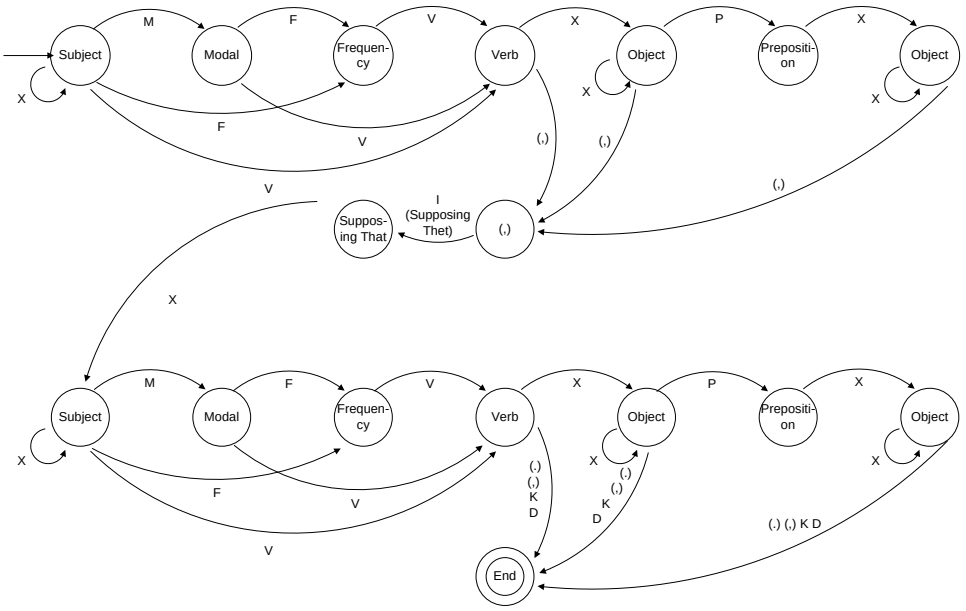


Gambar 3.40. *Syntax* 2 kata "*Assuming that*".

c. *Syntax* kata "*Supposing that*".

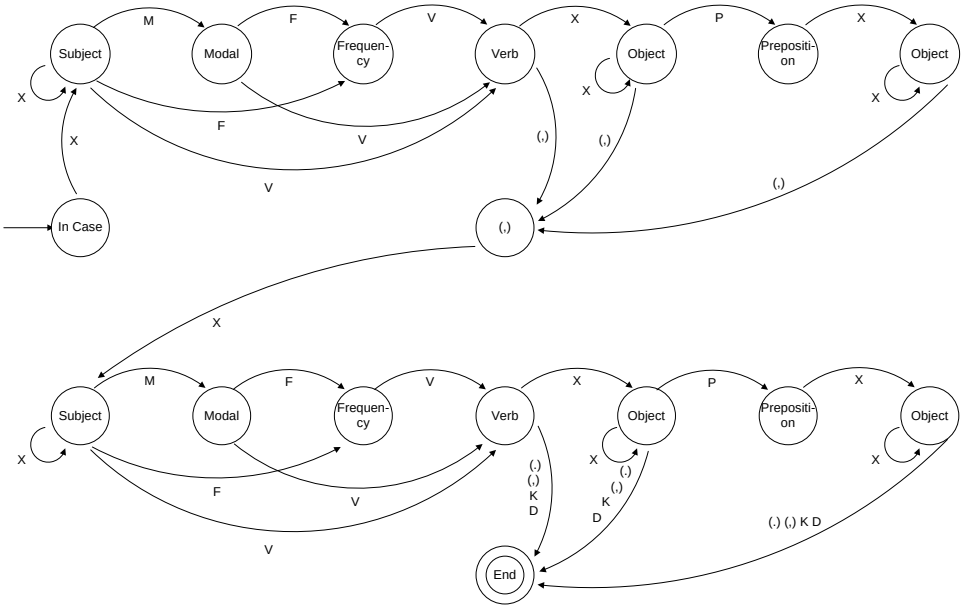


Gambar 3.41. *Syntax* 1 kata "*Supposing that*".

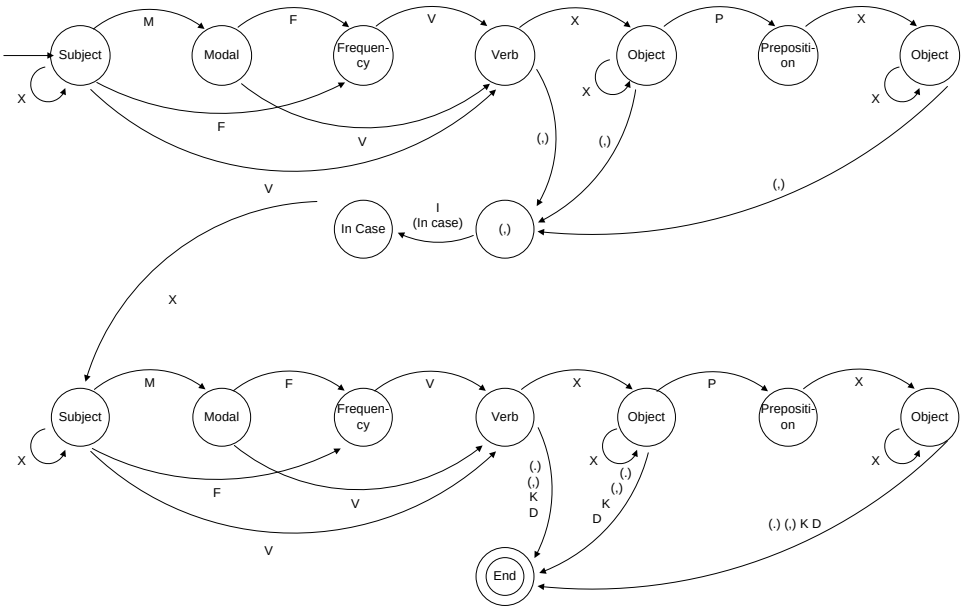


Gambar 3.42. *Syntax* 2 kata "*Supposing that*".

d. *Syntax* kata "*In case*".

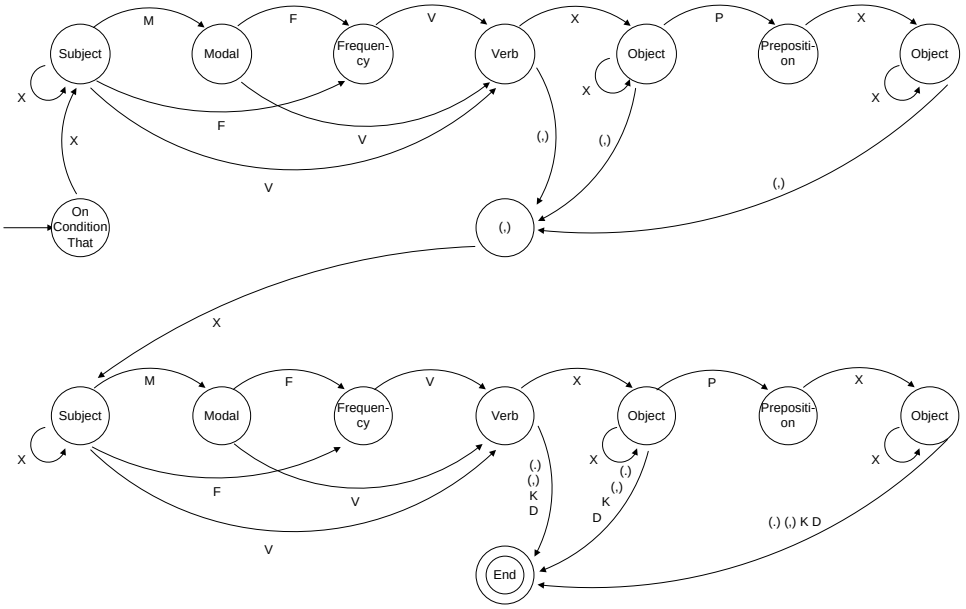


Gambar 3.43. *Syntax* 1 kata "*In case*".

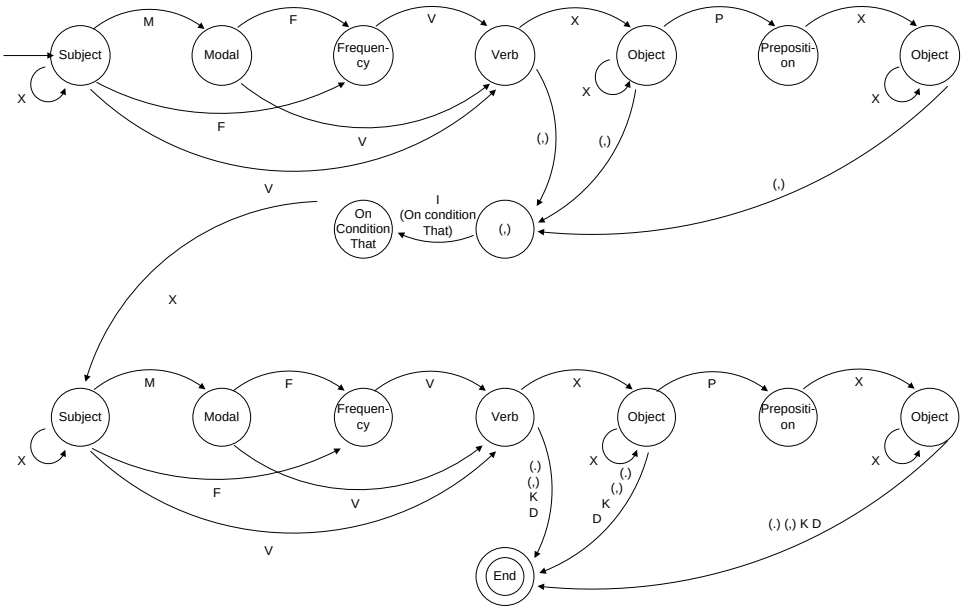


Gambar 3.44. *Syntax* 2 kata "*In case*".

e. *Syntax* kata "*On condition that*".

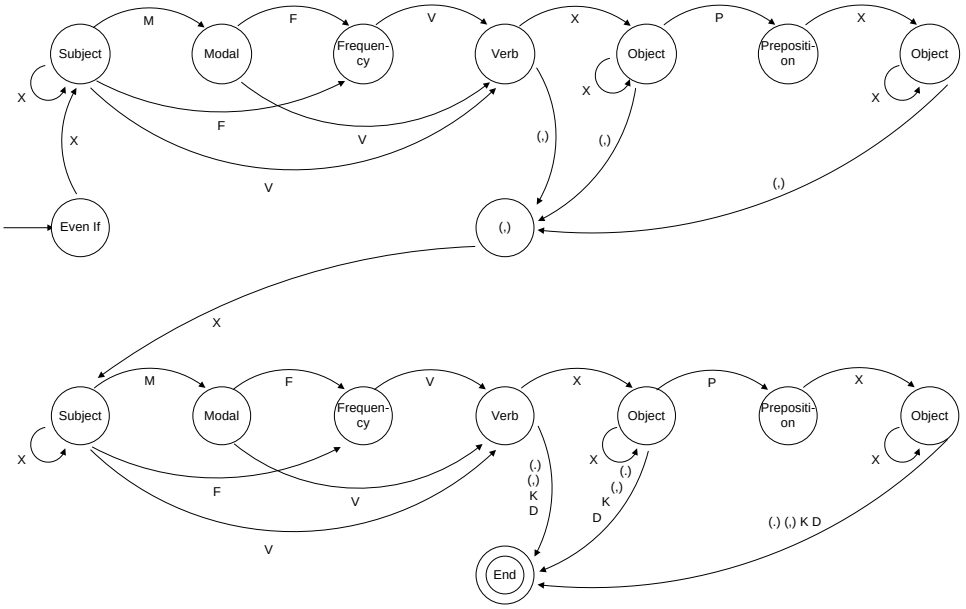


Gambar 3.45. *Syntax* 1 kata "*On condition that*".

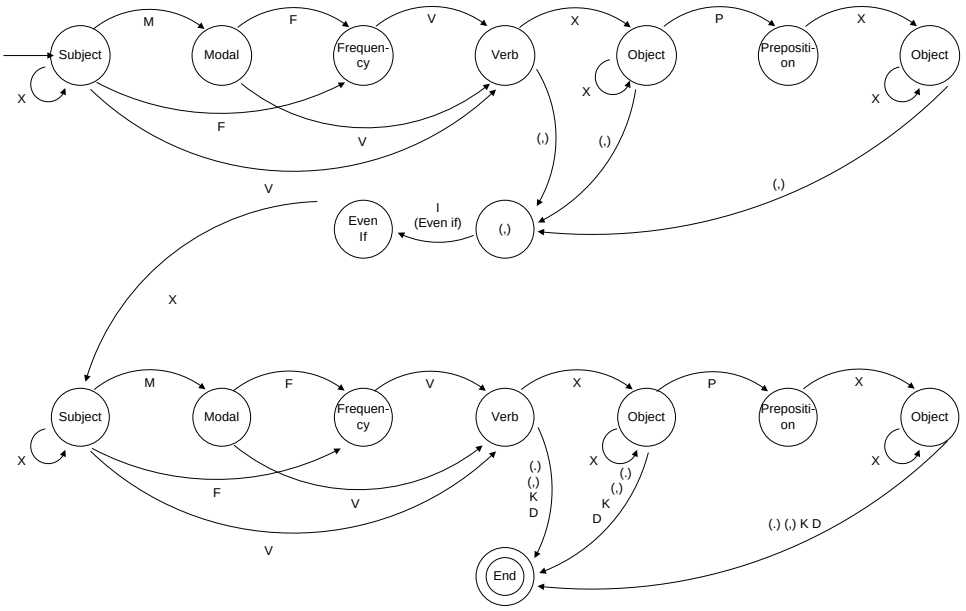


Gambar 3.46. *Syntax* 2 kata "*On condition that*".

f. *Syntax* kata "*Even if*".

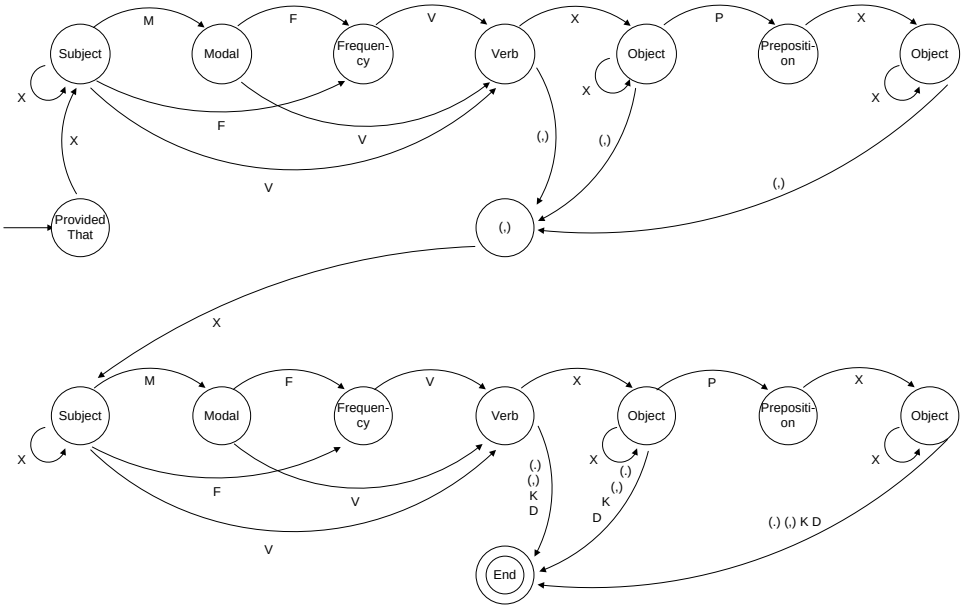


Gambar 3.47. *Syntax* 1 kata "*Even if*".

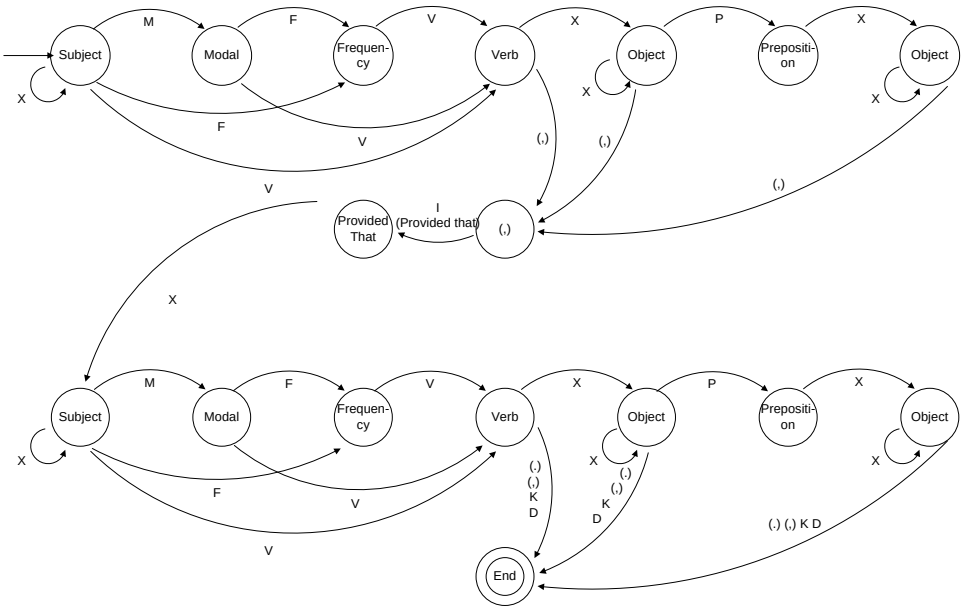


Gambar 3.48. *Syntax* 2 kata "*Even if*".

g. *Syntax* kata "*Provided that*".

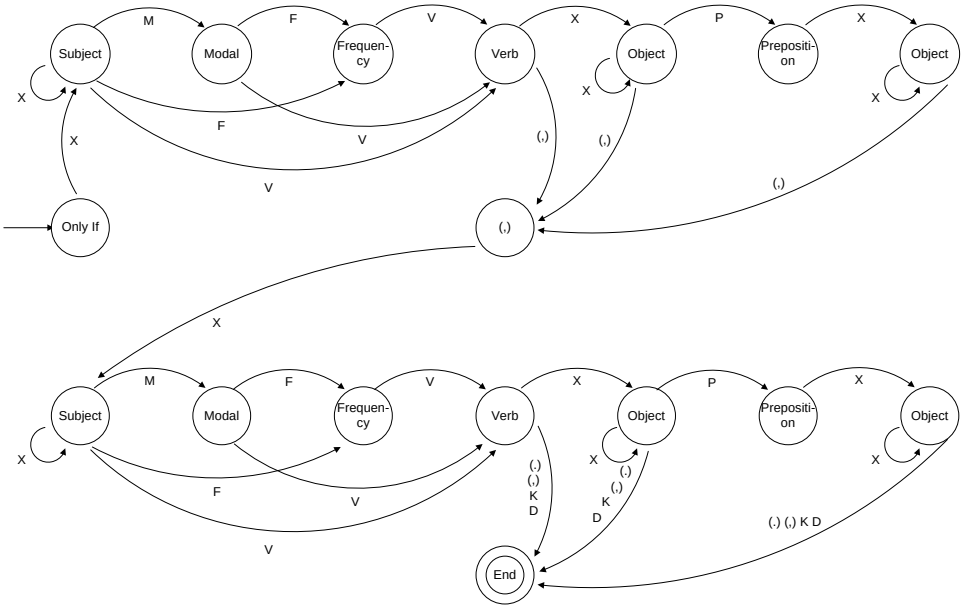


Gambar 3.49. *Syntax* 1 kata "*Provided that*".

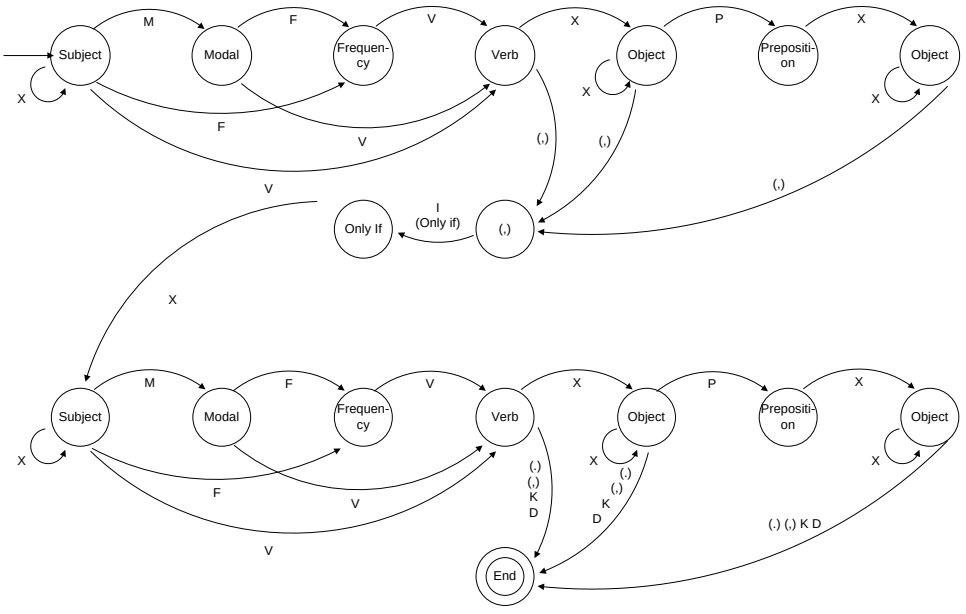


Gambar 3.50. *Syntax* 2 kata "*Provided that*".

h. *Syntax* kata "*Only if*".



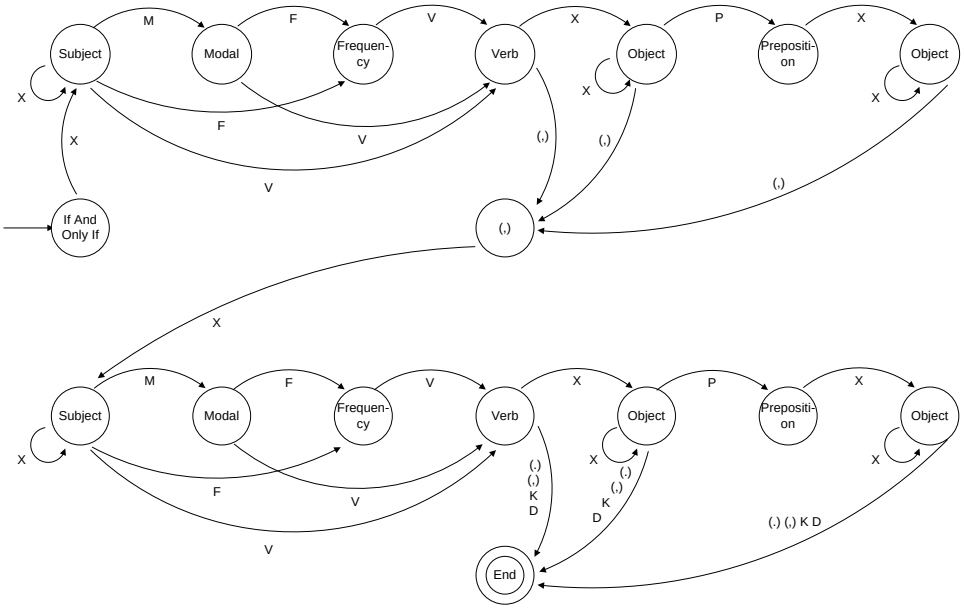
Gambar 3.51. *Syntax* 1 kata "*Only if*".



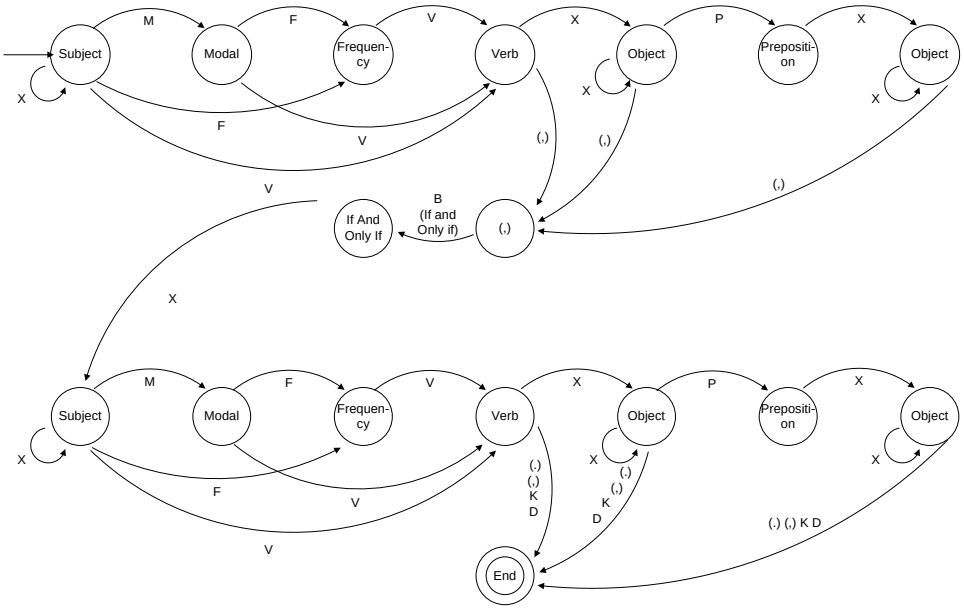
Gambar 3.52. *Syntax* 2 kata "*Only if*".

3.7.5. Konektif Biimplikasi.

a. *Syntax* kata "*If and only if*".

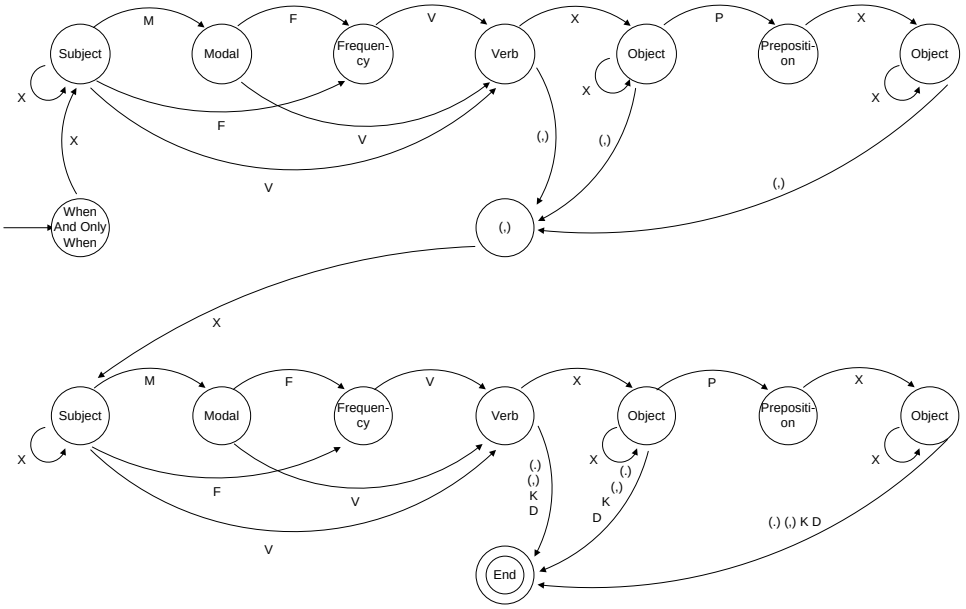


Gambar 3.53. *Syntax* 1 kata "*If and only if*".

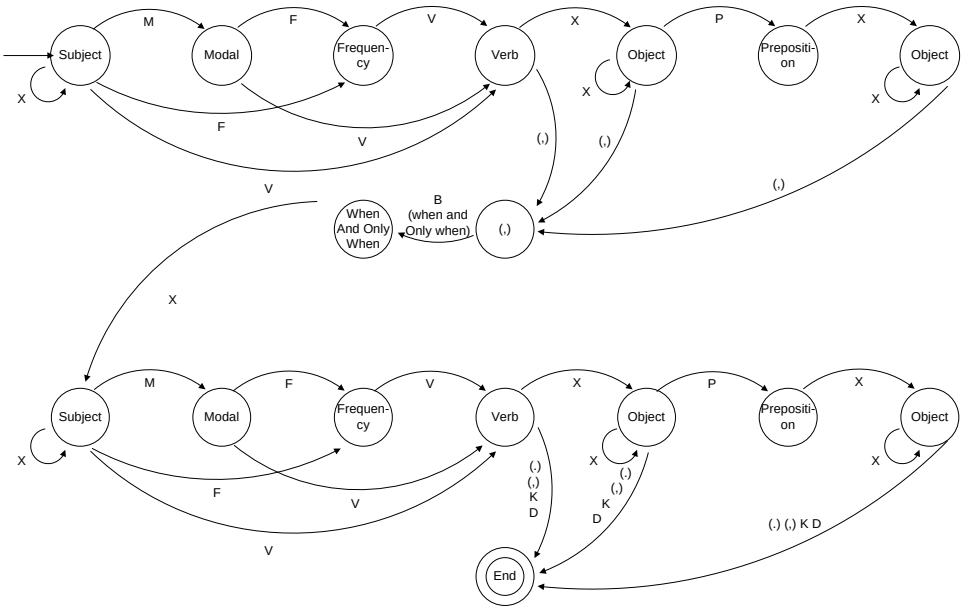


Gambar 3.54. *Syntax* 2 kata "*If and only if*".

b. *Syntax* kata "*When and only when*".

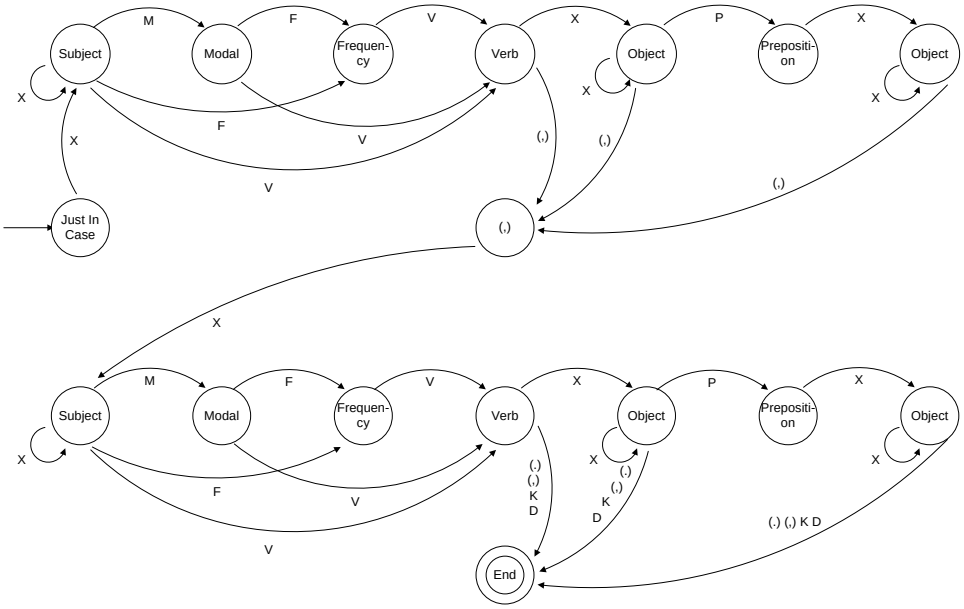


Gambar 3.55. *Syntax* 1 kata "*When and only when*".

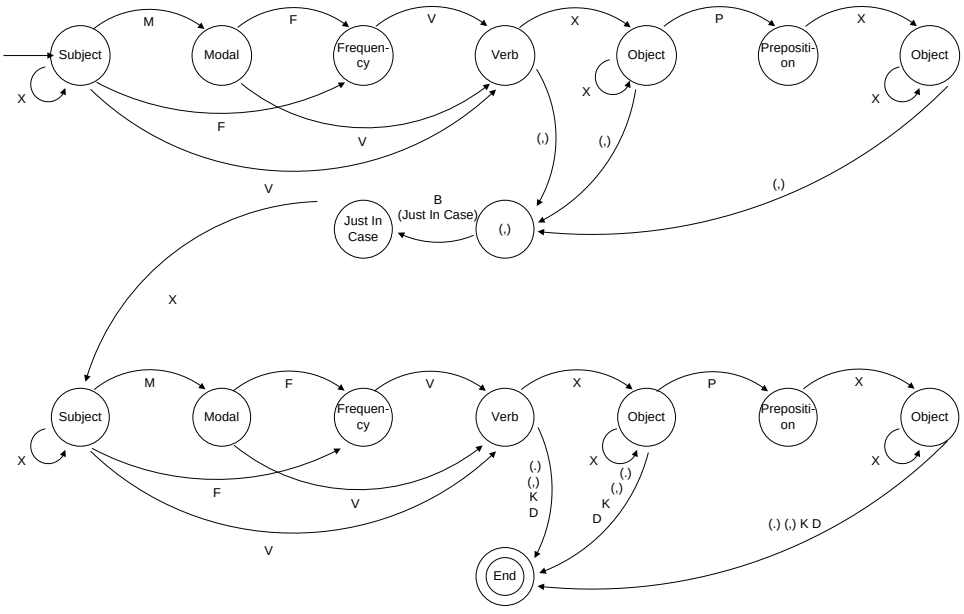


Gambar 3.56. *Syntax* 2 kata "*When and only when*".

c. *Syntax* kata "*Just in case*".



Gambar 3.57. *Syntax* 1 kata "*Just in case*".



Gambar 3.58. *Syntax* 2 kata "*Just in case*".