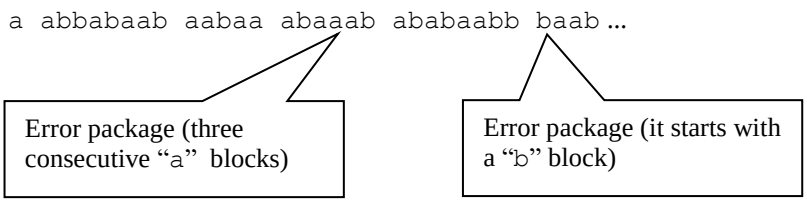
UNIT 6: BOTTOM-UP PARSING TECHNIQUES

We want to develop an analyzer that can verify that the data packages that circulate through an information channel have the appropriate structure.

A data package consists of two types of blocks, the blocks labeled with the letter "a" and those labeled with the letter "b". Valid data packages have the following structure:

- They start with a block "a"
- They can have any number of blocks greater than or equal to 1.
- They cannot have more than two consecutive blocks with the same label.

Example:



1. Define the grammar G of the analyzer.

2. Construct an SLR(1) parsing table for the analyzer.

1.-

The problem of generating the grammar that will be used to perform the parsing can be approached from two points of view:

- Make a grammar that accepts only the valid words of the language. It has the advantage of not requiring any additional control, but the drawback of making recovery of the error more difficult and giving information about it.
- Construct a more general grammar that accepts valid and erroneous sentences, later by means of actions in the production rules, the syntactic analysis must be performed. It has the advantage of facilitating the recovery of errors and allows giving more information about the reason for the error.

Both approaches are valid, in this solution the second one will be developed.

A general grammar, the symbol E is the axiom, which accepts valid information packets (also accepts erroneous) is:

E::= S d E | S
S::=a S | b S | a | b

Factoring:

E::= S E'
E'::= d, E | λ
S::= a R | b T
R::= S | λ
T::= S | λ

The production rules for R and T are the same and can be simplified:

E::= S E'
E'::= d E | λ
S::= a R | b R
R::= S | λ

The inclusion of the token "d," allows to separate the packages, it is the token that represents the blank spaces.

2.-

1. E''::= E
2. E'::= λ
3. E'::= d E
4. E::= S E'
5. R::= λ
6. R::= S
7. S::= a R
8. S::= b R

State 0	Action	Go To
E''::= ·E		[0,E]=1
E::= ·S E'		[0,S]=2
S::= ·a R	[0,a]=D3	
S::= ·b R	[0,b]=D4	
State 1	Action	Go To
E''::= E·	[1,$]=ACP	
State 2	Action	Go To
E::= S ·E'		[2,E']=5
E'::= λ	[2,$]=R2	
E'::= ·d E	[2,d]=D6	
State 3	Action	Go To
S::= a ·R		[3,R]=7
R::= ·λ	[3,d]=R5	
	[3,$]=R5	
R::= ·S		[3,S]=8
S::= ·a R	[3,a]=D3	
S::= ·b R	[3,b]=D4	
State 4	Action	Go To
S::= b ·R		[4,R]=9
R::= ·λ	[4,d]=R5	
	[4,$]=R5	
R::= ·S		[4,S]=8
S::= ·a R	[4,a]=D3	
S::= ·b R	[4,b]=D4	
State 5	Action	Go To
E::= S E'·	[5,$]=R4	
State 6	Action	Go To
E'::= d ·E		[6,E]=10
E::= ·S E'		[6,S]=2
S::= ·a R	[6,a]=D3	
S::= ·b R	[6,b]=D4	
State 7	Action	Go To
S::= a R·	[7,d]=R7	
	[7,$]=R7	
State 8	Action	Go To
R::= S·	[8,d]=R6	
	[8,$]=R6	
State 9	Action	Go To
S::= b R·	[9,d]=R8	
	[9,$]=R8	
State 10	Action	Go To
E'::= d E·	[10,$]=R3	

Table SLR(1)		**Shift**				**Go To**			
		$	**a**	**b**	**d**	**E'**	**E**	**R**	**S**
States	**0**		D3	D4			1		2
	1	ACP							
	2	R2			D6	5			
	3	R5	D3	D4	R5			7	8
	4	R5	D3	D4	R5			9	8
	5	R4							
	6		D3	D4			10		2
	7	R7			R7				
	8	R6			R6				
	9	R8			R8				
	10	R3							