

Analyse Syntaxique

Mohammed Akram RHAFRANE - Mehdi BOUTCHICHE
Nathanaël BERTRAND - Ismail SENHAJI

Encadrant: Martin Strecker

Université de Toulouse III/IRIT

Année 2012/2013

Plan

- 1 Objectifs du TER
- 2 Fondamentaux
- 3 Comparaison entre les outils
- 4 Xtext
- 5 Conclusion

Objectif du TER

- Initiation à la recherche bibliographique
- Rédaction :
 - Rapport sur le sujet (40 pages)*
 - Rapport du TER (10 pages)*
 - Présentation du TER*
- Tous les livrables sont rédigés en Latex

Plan

- 1 Objectifs du TER
- 2 Fondamentaux**
- 3 Comparaison entre les outils
- 4 Xtext
- 5 Conclusion

Plan

2 Fondamentaux

- Contexte
- Analyse Lexicale
- Analyse Syntaxique
- Analyse LL
- Analyse LR

Contexte

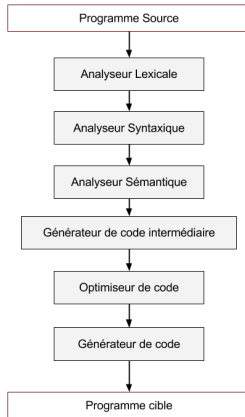


Figure : Processus de compilation

Plan

2 Fondamentaux

- Contexte
- **Analyse Lexicale**
- Analyse Syntaxique
- Analyse LL
- Analyse LR

Analyse Syntaxique

Principe

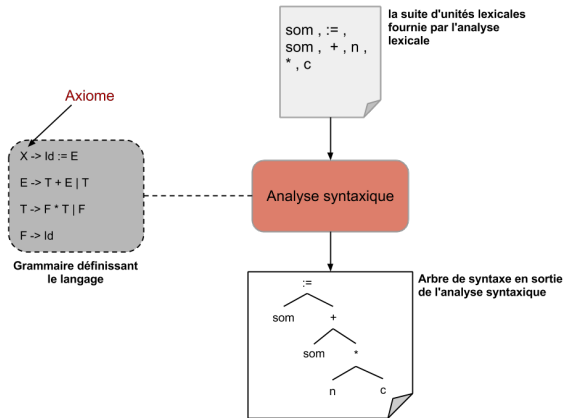


Figure : Analyse lexicale

Analyse Syntaxique

Exemple

$S \Rightarrow S ; S$

$S \Rightarrow id := E$

$S \Rightarrow \text{print}(L)$

$E \Rightarrow id$

$E \Rightarrow \text{num}$

$E \Rightarrow E + E$

$E \Rightarrow (S, E)$

$L \Rightarrow E$

$L \Rightarrow L, E$

$id := \text{num}; id := id + (id := \text{num} + \text{num}, id)$

Analyse Syntaxique

Exemple

S
S ; S
S ; id := E
id := E ; id := E
id := num ; id := E
id := num ; id := E + E
id := num ; id := E + (S , E)
id := num ; id := id + (S , E)
id := num ; id := id + (id := E , E)
id := num ; id := id + (id := E + E , E)
id := num ; id := id + (id := E + E , id)
id := num ; id := id + (id := num + E , id)
id := num ; id := id + (id := num + num , id)

Plan

2 Fondamentaux

- Contexte
- Analyse Lexicale
- **Analyse Syntaxique**
- Analyse LL
- Analyse LR

Analyse LL

Définition

- Left to right
- Leftmost derivation
- $LL(k)$

Analyse LL

Exemple

mot analysé: $\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \text{num} + \text{num}, \text{id})$

S

S ; S

$\text{id} := \underline{\text{E}} ; \text{S}$

$\text{id} := \text{num} ; \underline{\text{S}}$

$\text{id} := \text{num} ; \text{id} := \underline{\text{E}}$

$\text{id} := \text{num} ; \text{id} := \underline{\text{E}} + \text{E}$

$\text{id} := \text{num} ; \text{id} := \text{id} + \underline{\text{E}}$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\underline{\text{S}}, \text{E})$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \underline{\text{E}}, \text{E})$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \underline{\text{E}} + \text{E}, \text{E})$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \text{num} + \underline{\text{E}}, \text{E})$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \text{num} + \text{num}, \underline{\text{E}})$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \text{num} + \text{num}, \text{id})$

Plan

2 Fondamentaux

- Contexte
- Analyse Lexicale
- Analyse Syntaxique
- **Analyse LL**
- Analyse LR

Analyse LR

Définition

- Left to right
- Rightmost derivation
- LR(k)

Analyse LR

Exemple

mot analysé: $\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \text{num} + \text{num}, \text{id})$

S

S ; S

S ; $\text{id} := \underline{\text{E}}$

S ; $\text{id} := \text{E} + \underline{\text{E}}$

S ; $\text{id} := \text{E} + (\text{S}, \underline{\text{E}})$

S ; $\text{id} := \text{E} + (\underline{\text{S}}, \text{id})$

S ; $\text{id} := \text{E} + (\text{id} := \underline{\text{E}}, \text{id})$

S ; $\text{id} := \text{E} + (\text{id} := \text{E} + \underline{\text{E}}, \text{id})$

S ; $\text{id} := \text{E} + (\text{id} := \underline{\text{E}} + \text{num}, \text{id})$

S ; $\text{id} := \underline{\text{E}} + (\text{id} := \text{num} + \text{num}, \text{id})$

S ; $\text{id} := \text{id} + (\text{id} := \text{num} + \text{num}, \text{id})$

$\text{id} := \underline{\text{E}} ; \text{id} := \text{id} + (\text{id} := \text{num} + \text{num}, \text{id})$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \text{num} + \text{num}, \text{id})$

Analyse LR

Types d'analyseurs LR

- Analyseurs LR simples (SLR)
- Analyseurs syntaxiques LR avec anticipation (LALR)
- Analyseurs syntaxiques canoniques

Plan

- 1 Objectifs du TER
- 2 Fondamentaux
- 3 Comparaison entre les outils**
- 4 Xtext
- 5 Conclusion

Plan

3 Comparaison entre les outils

- YACC
- ANTLR
- Résultat

YACC

Définitions

- Génération d'analyseur syntaxique
A partir d'une spécification
- Langage : C
Les analyseurs syntaxique généré par YACC sont en langage C
- Type LALR
Les analyseurs syntaxique généré par YACC sont de type LALR

YACC

Description de YACC

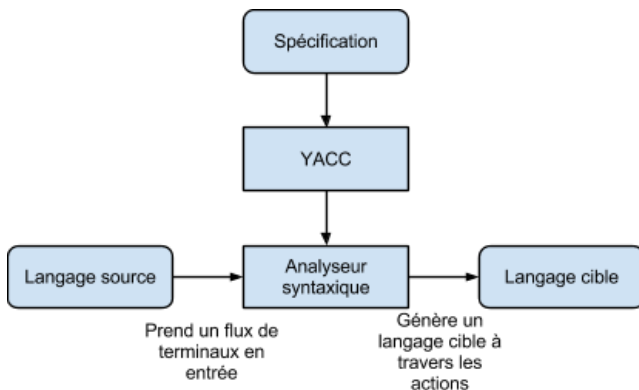


Figure : Description de YACC

YACC

Spécifications

Les spécifications permettent:

- Description du langage
- Spécification des actions associées aux règles de grammaires
- Génération d'analyseur syntaxique

YACC I

Structure de la spécification

La spécification suit la structure suivante :

Bloc des déclarations

%%

Règles de grammaire

%%

Programme

Règles de grammaire :

A : B { /* action pour cette regle */ };

YACC

Exemple

- Exemple de spécification YACC *Génération d'une calculatrice*
- But: Intérpréteur de commande permettant de faire des opération telles que:
 $2+5$
 $(1+2)$
 $1*2$
 $2*(2+1)$
- Plus de simplicité:
les nombres à plusieurs chiffres ne sont pas pris en charge

YACC I

Exemple

$G(L) = \langle N, X, P, S \rangle$ Avec

$N : \{ \text{ligne, commande, expr, terme, facteur} \}$

$X : \{ [0-9], \backslash n, +, *, (,) \}$

```
P : {  
    ligne ->  
        commande \n ligne  
        | \n  
    commande :  
        expr  
    expr :  
        expr + terme  
        | terme  
    terme :  
        terme * facteur  
        | facteur  
    facteur :  
        ( expr )  
        | [0-9]  
}  
S : ligne
```

YACC I

Exemple

Spécification d'une mini calculatrice :

```
%{  
    #include <ctype.h>  
    #include <stdio.h>  
    #include <stdlib.h>  
%}  
  
%token CHIFFRE  
  
%%
```

YACC I

Exemple

```
ligne : commande '\n' ligne
      | '\n' { printf("Fin du programme\n"); exit(0); }
      ;
```

```
commande: expr { printf("Resultat: %d\n", $1); };
```

```
expr : expr '+' terme { $$ = $1 + $3; }
      | terme
      ;
```

```
terme : terme '*' facteur { $$ = $1 * $3; }
       | facteur
       ;
```

```
facteur : '(' expr ')' { $$ = $2; }
         | CHIFFRE
         ;
```

```
%%
```

YACC I

Exemple

```
int main(){
    yyparse();
}

int yyerror(char *s){
    printf("%s \n", s);
}

int yylex(){
    int c;
    c = getchar();
    if(isdigit(c)){
        yylval = c-'0';
        return CHIFFRE;
    }
    return c;
}
```

YACC

Spécifications

Pour faire fonctionner cet exemple, il faudra entrer les lignes de commande suivantes:

- `> bison exemple1.y`
- `> gcc exemple1.tab.c -o exemple1`
- `> ./exemple1`

Plan

3 Comparaison entre les outils

- YACC
- ANTLR
- Résultat

Antlr

Definition

- - Générateur de parseur public
- - LL(k)
- - Framework

Antlr

Quelques Fonctionnalités

- intègre la spécification entre une analyse lexicale et syntaxique.
- facilite la construction de l'arbre syntaxique.
- génère des parseurs de descente récursives en C et C++.
- facilite la gestion d'erreurs.

Antlr

Elements du langage:

Elément de langages	Description	Exemple
Token	Commence par majuscule	ID
<code><< ... >></code>	Définie une action sémantique	<code><< printf ("%S", a); >></code>
<code>(...)</code>	Règle	<code>("int" ID storage_class)</code>
<code>(...)*</code>	Closure	<code>ID (" " ID)*</code>
<code>(...)+</code>	Positive Closure	<code>slist : (stat SEMICOLON)</code>
<code>{...}</code>	Optionnel	<code>{ELSE stat}</code>
<code><< ... >>?</code>	Prédicat sémantique	<code>type : << is_Type (str) >>? ID</code>
<code>(...)?</code>	Prédicat syntaxique	<code>((list EQ)? list EQ list list</code>

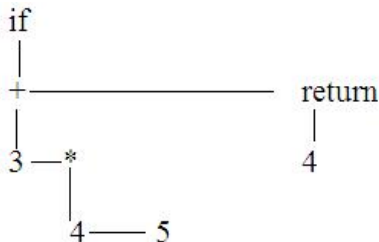
Antlr I

Gestion de l'arbre

Voici un petit exemple de la construction d'un arbre syntaxique :

```
if 3 + 4 * 5 then return 4;
```

L'arbre correspondant :



Antlr

Gestion d'erreurs

Deux mécanismes de gestion d'erreurs:

- Gestion automatique d'erreurs
- *Parser Exception Handling*

Antlr I

Exemple de grammaire Antlr: Calculatrice simple

```
grammar GrammaireSimple;

options {
    language = Java;
}

tokens {
    PLUS      = '+' ;
    MINUS     = '-' ;
    MULT      = '*' ;
    DIV = '/' ;
}

@members {
public static void Main(string[] args) {
    GrammaireSimpleLexer lex = new GrammaireSimpleLexer(new ANTLRFileStream(args[0]));
    CommonTokenStream tokens = new CommonTokenStream(lex);

    GrammaireSimplecParser parser = new GrammaireSimpleParser(tokens);
```

Antlr II

Exemple de grammaire Antlr: Calculatrice simple

```
try {  
parser.expr();  
} catch (RecognitionException e) {  
Console.Error.WriteLine(e.StackTrace);  
}  
}
```

%PARSER RULES

```
expr      : term ( ( PLUS | MINUS ) term )* ;  
  
term      : factor ( ( MULT | DIV ) factor )* ;  
  
factor    : NUMBER ;
```

%LEXER RULES

```
NUMBER    : (DIGIT)+ ;
```

Antlr III

Exemple de grammaire Antlr: Calculatrice simple

```
WHITESPACE : ( '\t' | ' ' | '\r' | '\n' | '\u000C' )+    { $channel = Hidden; } ;  
fragment DIGIT : '0'..'9' ;
```

Plan

3 Comparaison entre les outils

- YACC
- ANTLR
- Résultat

Résultat

Yacc(LALR) vs Antlr(LL):

- Récursivité à gauche
- complexité d'utilisation
- Performance et flexibilité

Plan

- 1 Objectifs du TER
- 2 Fondamentaux
- 3 Comparaison entre les outils
- 4 Xtext**
- 5 Conclusion

Xtext

Définition

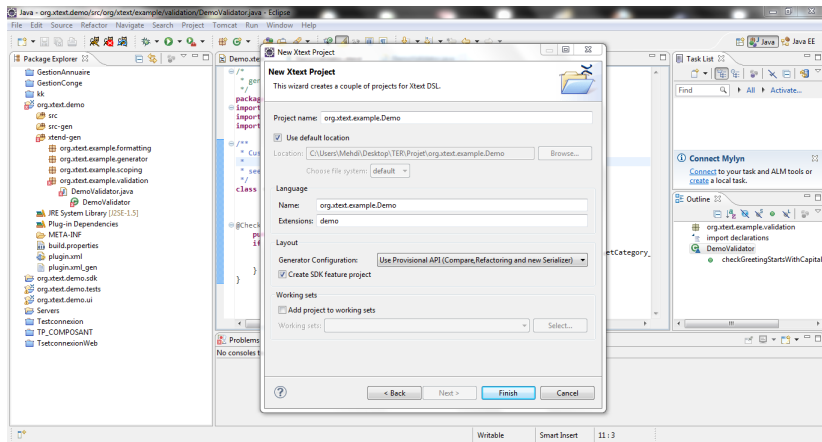
- - Framework Eclipse
- - Développement de langages de programmation et de DSL
- - Grammaire proche de celle de Antlr

Parmi les fonctionnalités qu'offre Xtext:

- -coloration syntaxique
- -auto complétion
- -validation
- -intégration avec d'autres composants Eclipse

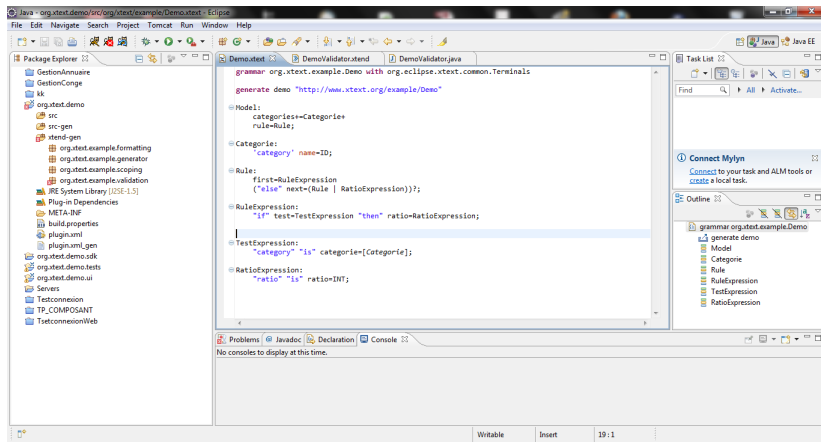
Xtext:Exemple

Etape 1: Création d'un projet Xtext



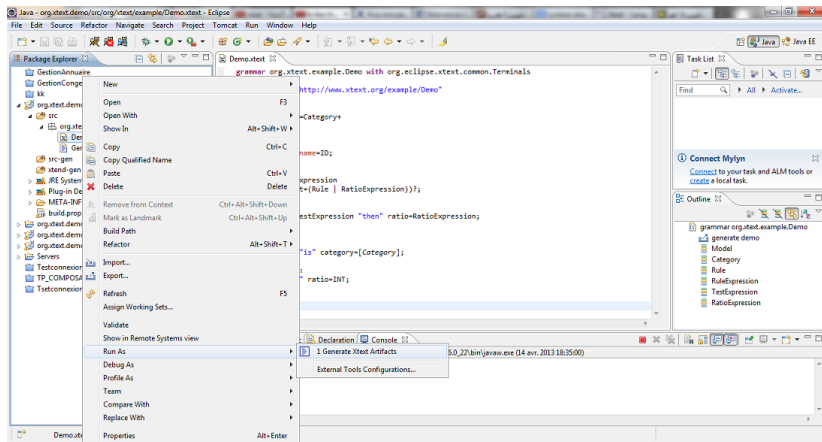
Xtext: Exemple

Etape 2: Definition du langage



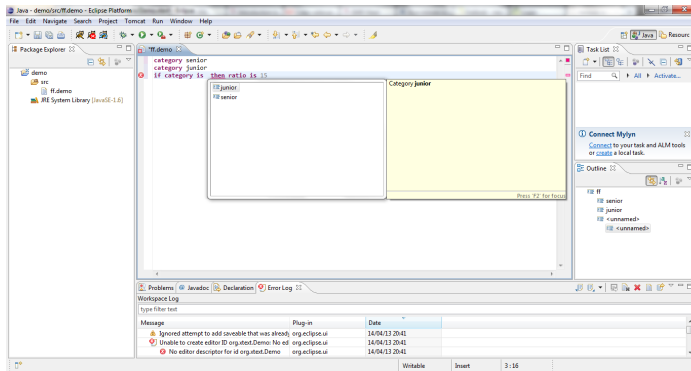
Xtext: Exemple

Etape 3: Generation des artefacts



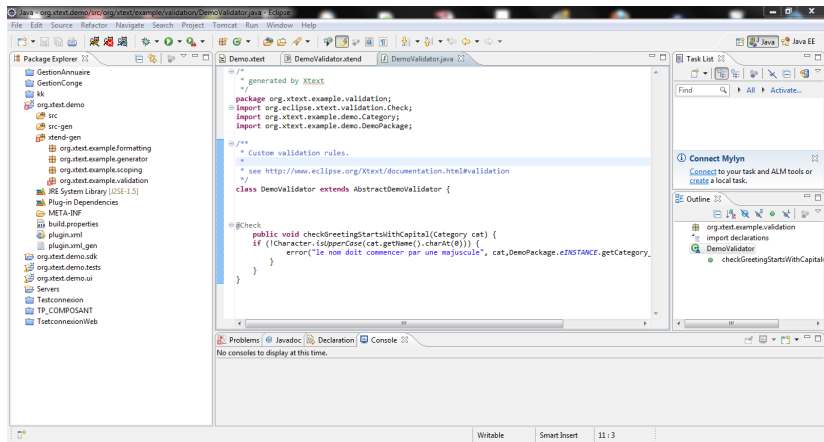
Xtext:Exemple

Etape 4: Faire un test



Xtext: Exemple

Etape 5: Ajouter une validation



Plan

- 1 Objectifs du TER
- 2 Fondamentaux
- 3 Comparaison entre les outils
- 4 Xtext
- 5 Conclusion**

Conclusion

- Initiation au travail de recherche
- Qualité de rédaction
- Organisation au sein d'une équipe de recherche

