ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

Avenida Professor Luciano Gualberto, travessa 3 nº 158 CEP 05508-900 São Paulo SP
Telefone: (011) 818-5583 Fax (011) 818-5294

Departamento de Engenharia de Computação e Sistemas Digitais

Linguagens e Compiladores - Compilador Léxico

Felipe Giunte Yoshida N^{o}USP 4978231
Mariana Ramos Franco N^{o}USP 5179364

24 de Setembro de 2009

1 Introdução

A função básica desta etapa do compilador é receber o código fonte e dividi-lo em tokens. Para tal, foi sugerido pelo professor a criação de um autômato finito determinístico (AFD), que recebe caracter por caracter o código fonte, o "quebra" em tokens e monta uma tabela que identifica alguns tipos de token.

Inicialmente tentamos realizar a tarefa na linguagem C, mas devido a problemas de compatibilidade de bibliotecas entre Unix/Linux e Windows, resolvemos abandonar tal abordagem e reescrever o programa em Java, o que no final se mostrou mais organizado e prático.

2 Estruturas

Para facilitar a estruturação do código, criamos as seguintes estruturas:

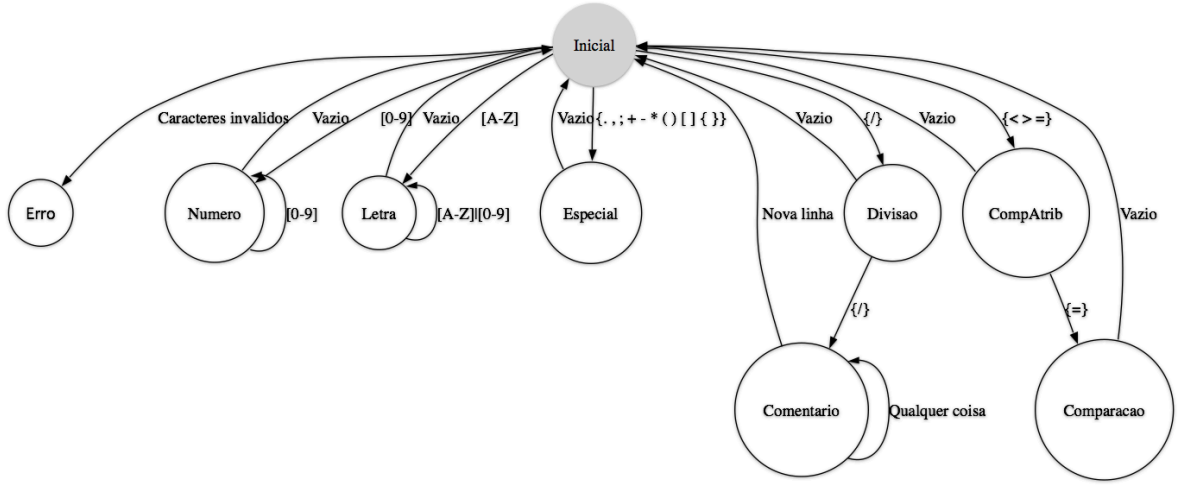
- AFD - Autômato finito determinístico, composto de estados e o estado ativo
- Estado - Estado e suas transições
- Transicao - Entradas que ativam uma transição e o estado que é ativado
- Token - Parte do código, que é classificada
- FluxoTokens - Seqüência de tokens
- TabelaSimbolos - Tabela auxiliar, usada na classificação dos tokens
- Simbolo - Entrada da tabela de símbolos

- Tipo - Classificação dos tipos de token
- PalavrasReservadas - Definição das palavras reservadas

3 Lógica

Podemos dividir a parte lógica desta etapa em basicamente duas partes:

Montador do autômato Descrito na classe *MontaAFD*, ele recebe um arquivo XML com a especificação do AFD abaixo, e o converte na estrutuda *AFD*.



Simulador do autômato Descrito na classe *PercorreAFD*, simula um AFD a partir do autômato montado no passo anterior e do código fonte, que é analisado caracter por caracter.