Um arcabouço para extensibilidade em linguagens de programação

Paulo Roberto Massa Cereda
paulo.cereda@usp.br

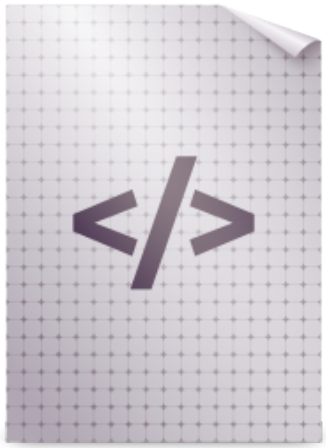
João José Neto
jjneto@usp.br

Escola Politécnica, Universidade de São Paulo

30 de Janeiro de 2015

WTA 2015

Motivação



- A área de pesquisa de linguagens de programação apresenta diversas contribuições extremamente significativas na utilização da tecnologia adaptativa em diversos níveis de abstração
- A possibilidade de extensão de linguagens de programação pode proporcionar o projeto e implementação de construtos especiais para a escrita de programas com características adaptativas

Nossa contribuição

- Um arcabouço que ofereça extensibilidade em linguagens de programação através da especificação de construtos sintáticos definidos pelo usuário em uma metalinguagem de extensão
  - Obter um programa-objeto escrito em linguagem extensível a ser executado pelo sistema operacional ou de um programa-fonte que transforma as extensões sintáticas em comando válidos da linguagem base

Linguagens extensíveis

- Linguagens de programação que permitem a adição ou alteração de construtos sintáticos ou a associação de novas formas sintáticas com semântica
- Novas notações ou operações, estruturas de controle novas ou modificadas, ou até mesmo elementos provenientes de diferentes paradigmas de programação

Notação de Wirth

Metalinguagem para descrição de linguagens de programação com uma notação simplificada em relação às iniciativas existentes

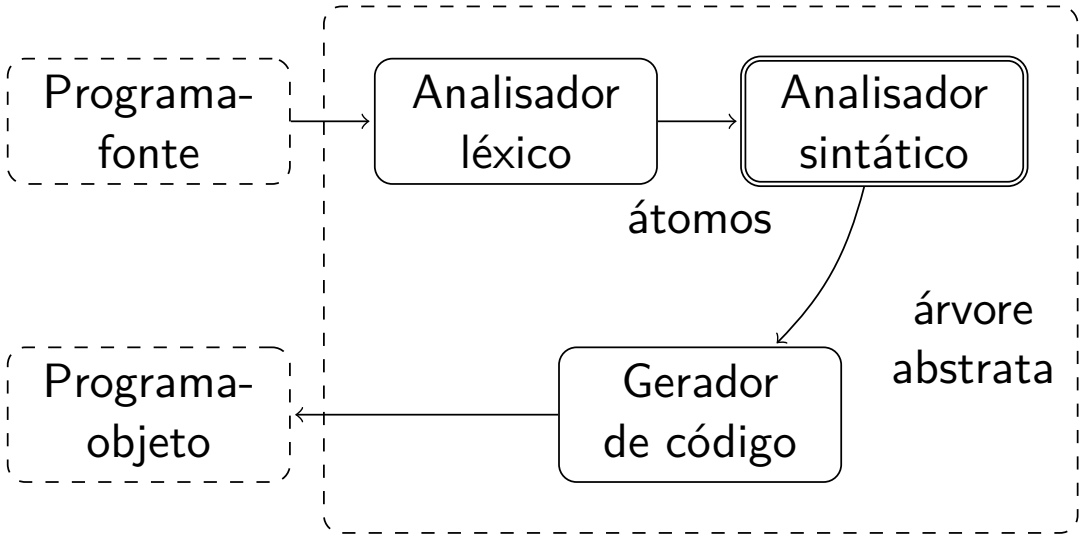
```
syntax = { production } .
production = identifier "=" expression "." .
expression = term { "|" term } .
term = factor { factor } .
factor = identifier | literal | "(" expression ")" |
            "[" expression "]" | "{" expression "}" .
literal = """" character { character } """" .
```

Notação de Wirth

1. Distinção clara entre metassímbolos, símbolos terminais e símbolos não-terminais
2. Não há restrição quanto à utilização de metassímbolos como símbolos da linguagem sendo descrita
3. Construto para iteração explícita.
4. Dispensa o uso de um símbolo explícito que represente a cadeia vazia
5. Utiliza o conjunto de caracteres ASCII

Analisador sintático

Deriva uma estrutura sintática para o programa a partir de um fluxo de tokens, adequando as palavras em um modelo gramatical da linguagem especificada



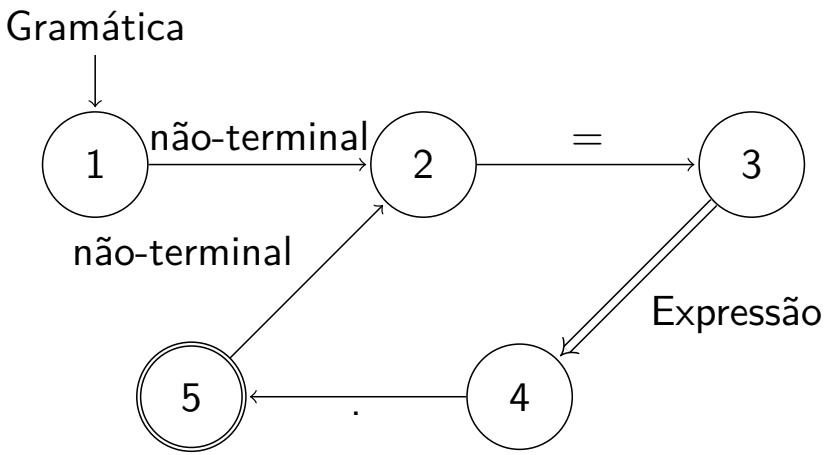
Geração de um analisador sintático

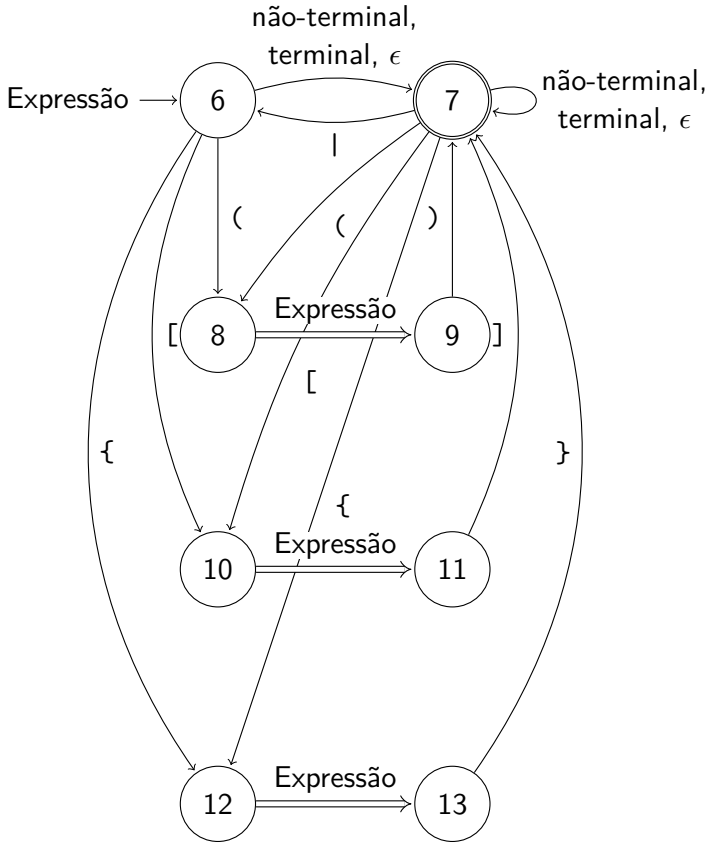
- É possível gerar um analisador sintático a partir da notação de Wirth, obtendo-se um autômato de pilha estruturado que reconheça sentenças válidas da gramática especificada

Requisitos:

1. a gramática esteja escrita na notação de Wirth
2. seja resolvido o sistema de equações representado pela gramática, interpretando os não-terminais como variáveis e os terminais como constantes

Geração de um analisador sintático



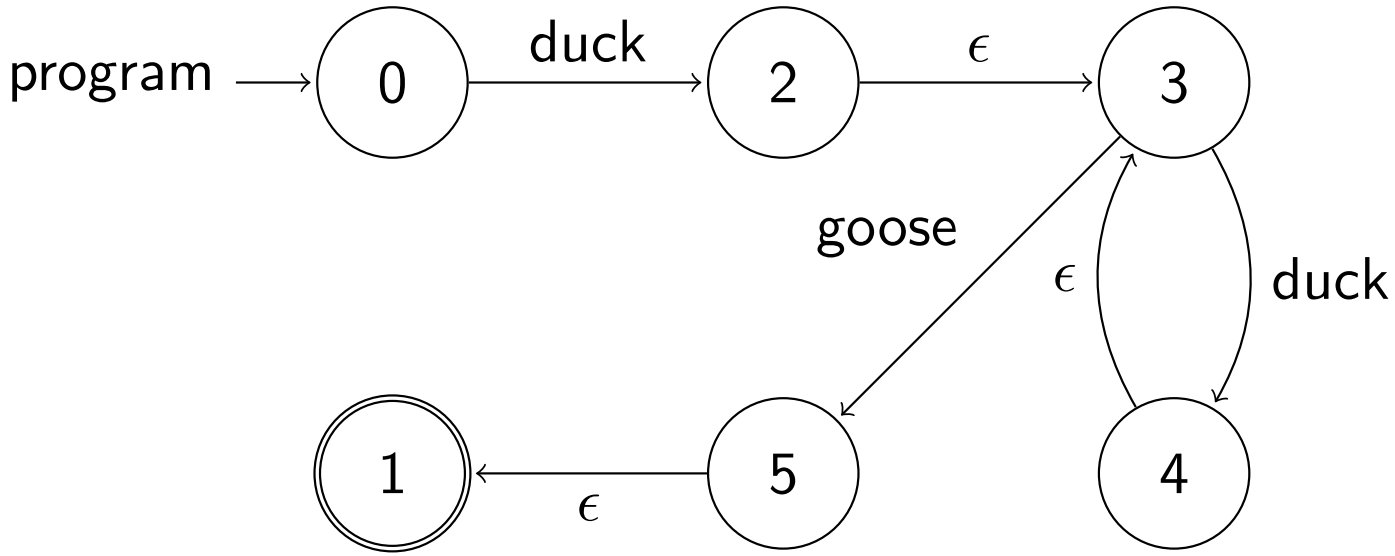


O autômato possui ações semânticas associadas às transições, permitindo a construção do analisador sintático.

Um exemplo didático

Gramática escrita em notação de Wirth

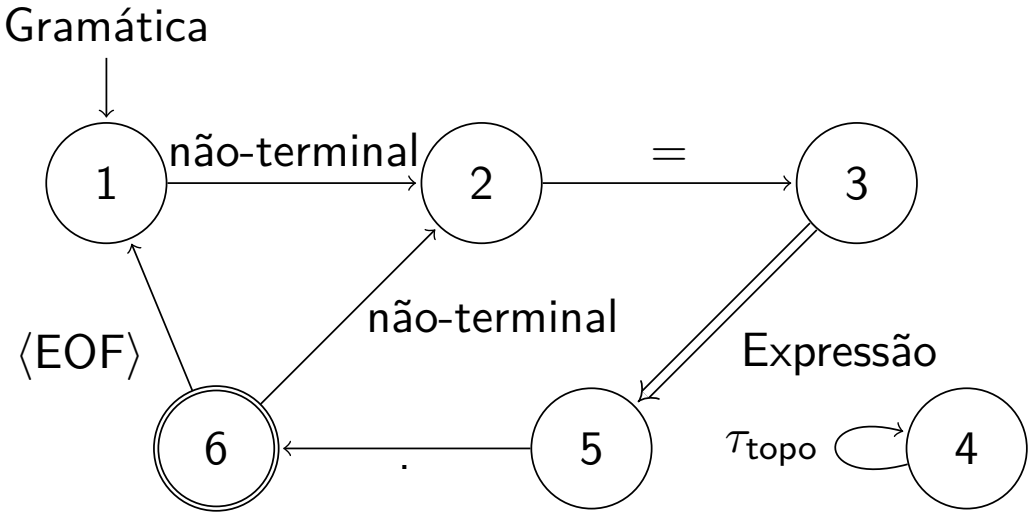
```
program = "duck" { "duck" } "goose" .
```



Introdução

- Arcabouço que ofereça extensibilidade em linguagens de programação, permitindo a adição ou a alteração de construtos sintáticos

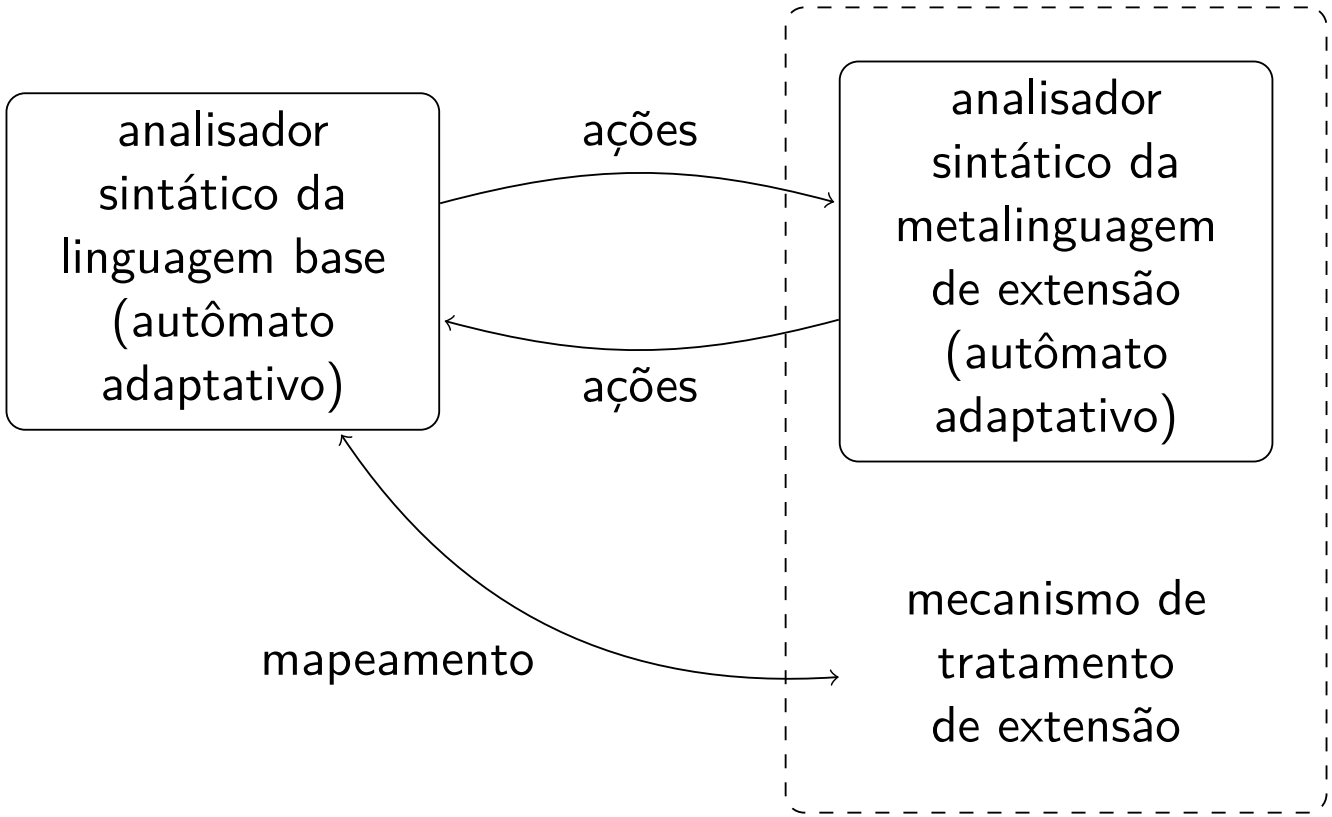
Autômato adaptativo para geração dos analisadores sintáticos (ações semânticas são substituídas por ações adaptativas) – a sub-máquina *Expressão* mantém-se inalterada.



Mecanismo de tratamento de extensões

- Manipula mudanças estruturais definidas pelo usuário, exigindo que o compilador modifique-se em tempo de compilação do programa-fonte, em resposta a tais mudanças requeridas sobre a linguagem original
- Analisa o processo de reconhecimento sintático da linguagem base, mapeando as novas construções sintáticas definidas na metalinguagem de extensão e disponíveis em seu componente sintático para o analisador original
- Uma metalinguagem é definida para o mecanismo de extensão, que realizará o mapeamento das novas construções sintáticas ao seu significado

Mecanismo de tratamento de extensões



Mecanismo de tratamento de extensões

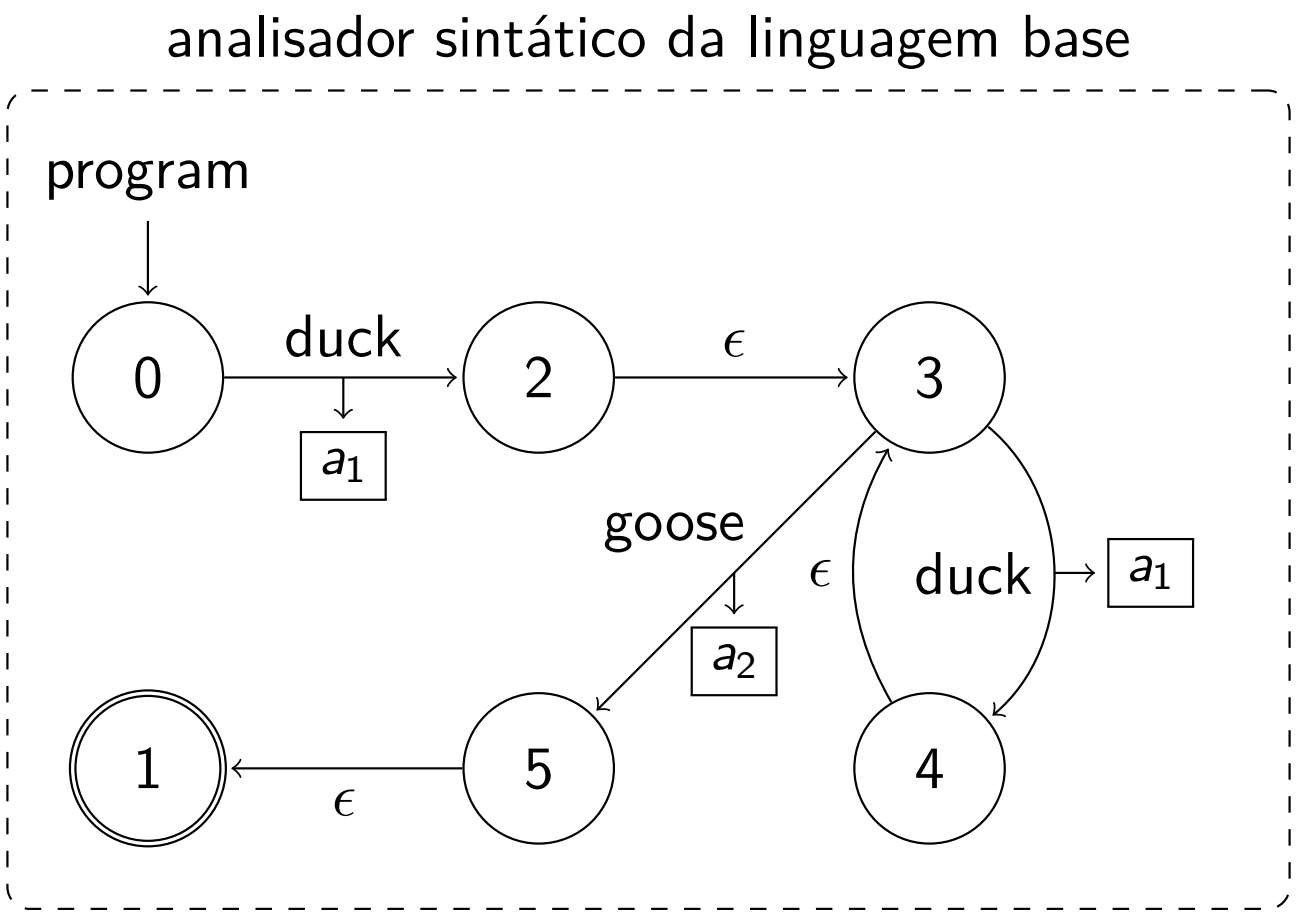
- O mecanismo de tratamento de extensões exerce um papel determinante para prover integração entre a linguagem básica e as extensões definidas pelo usuário
- A adequação da sintaxe estendida em tempo de execução é realizada através de ações adaptativas, que acomodam as novas construções no analisador sintático original

Um exemplo didático

Exemplo de metalinguagem hipotética

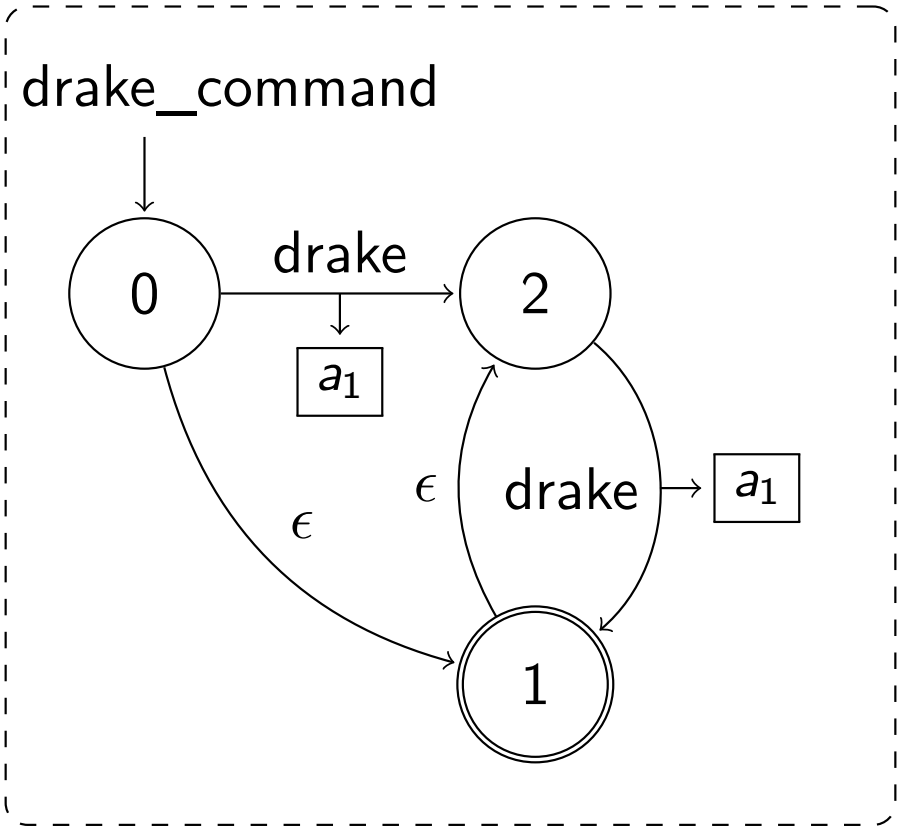
```
DEFINE drake_command AS
    [ "drake":1 { "drake":1 } ] .
INJECT drake_command AT
    program.
MEANING
    1:
        a1
END
```

Um exemplo didático

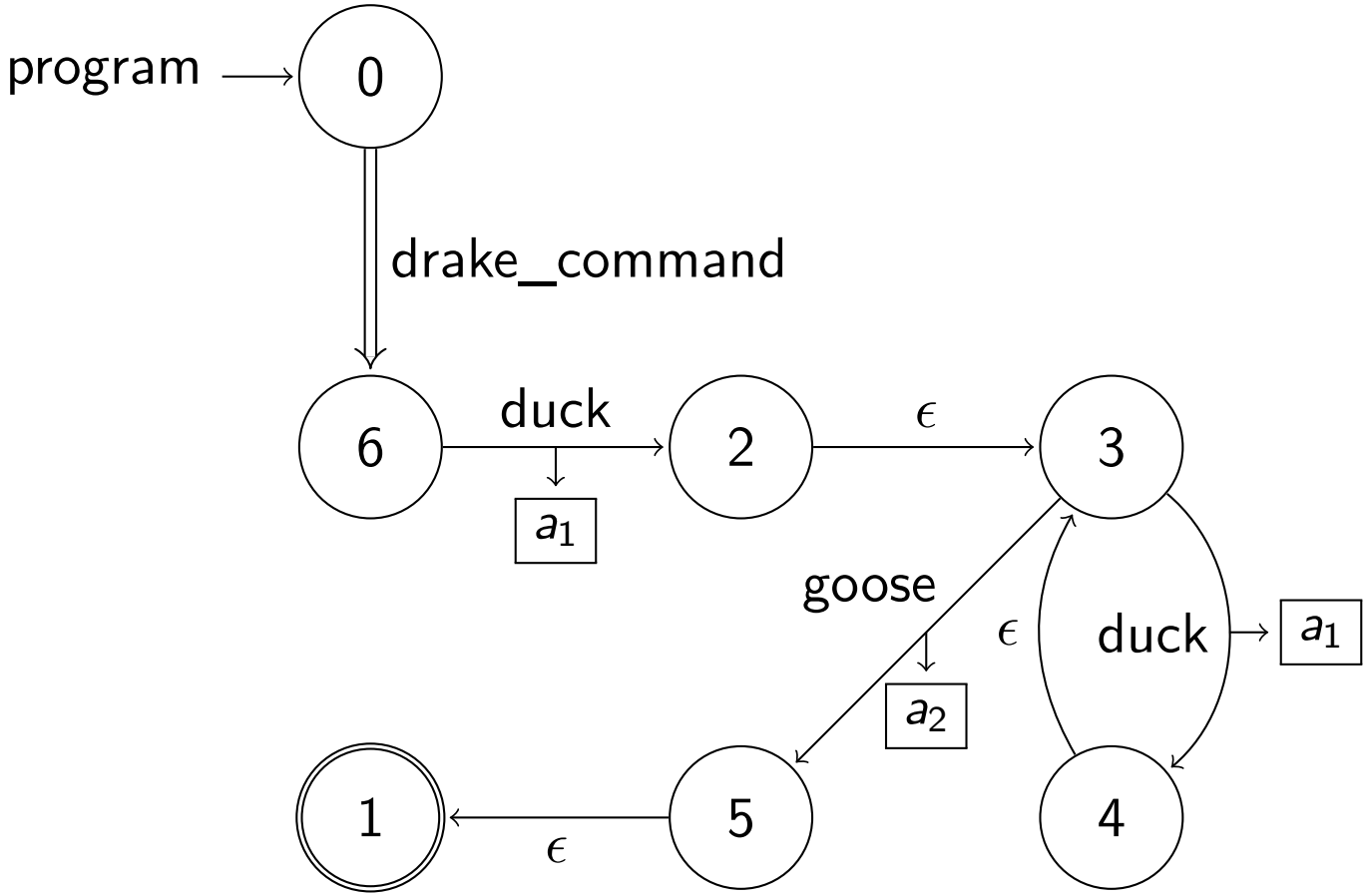


Um exemplo didático

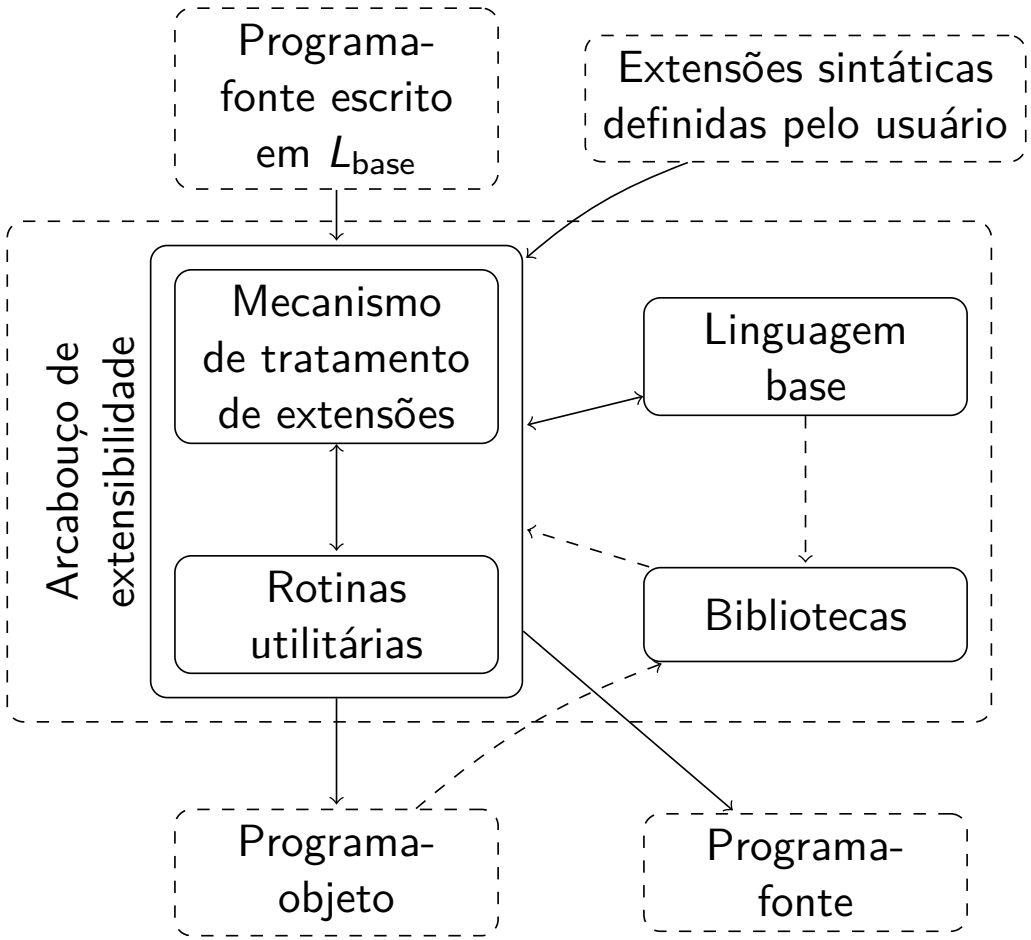
analisador sintático da metalinguagem de extensão



Um exemplo didático



Arquitetura do arcabouço



Arquitetura do arcabouço

- *Mecanismo de tratamento de extensões:*manipula mudanças estruturais definidas pelo usuário sobre a linguagem base
- *Rotinas utilitárias:* auxiliam o mecanismo de extensão durante o processo de reconhecimento e análise da linguagem base e das extensões definidas
- *Acoplamento da linguagem base:* mecanismos de comunicação para realizar a troca de informações entre o mecanismo de tratamento de extensão, as rotinas utilitárias e o próprio compilador da linguagem base
- *Bibliotecas:* bibliotecas do sistema operacional ou em nível do usuário e/ou adição de construtos adaptativos definidos em biblioteca de propósito geral

Um exemplo real

Pascal estendido

```
program test;
uses crt;
var
    a, b, c : integer;
begin
    a := 1; b := 2; c := 3;
    when (b > a) then begin
        a := b;
        when (b > c) then begin
            c := b;
        end;
    end;
    writeln(a); writeln(b);
    writeln(c);
end.
```

Pascal

```
program test;
uses crt;
var
    a, b, c : integer;
begin
    a := 1; b := 2; c := 3;
    if (b > a) then begin
        a := b;
        if (b > c) then begin
            c := b;
        end;
    end;
    writeln(a); writeln(b);
    writeln(c);
end.
```

Considerações finais

- O mecanismo de tratamento de extensão monitora a análise da linguagem base, realizando alterações de acordo com as extensões sintáticas definidas
- Permite obtenção de um programa-objeto a ser executado pelo sistema operacional ou de um programa-fonte que transforma as extensões sintáticas em comandos válidos da linguagem base

Considerações finais

- A extensibilidade é um poderoso recurso para a adição e alteração dos construtos sintáticos em linguagens de programação, aumentando seu poder expressivo
- Simplificação do processo de extensão de uma linguagem de programação seja simplificado de tal forma que o usuário possa beneficiar-se das novas notações, operações ou estruturas de controle para a escrita de programas mais consistentes
- Construtos especiais podem viabilizar o acréscimo de uma camada adaptativa em linguagens de programação tradicionais, permitindo a escrita de programas com características adaptativas

Obrigado!

Paulo Roberto Massa Cereda
paulo.cereda@usp.br

João José Neto
jjneto@usp.br