CFG, DFD, ...

Richard Bonichon

20140731

Sumário

Estilos de especificação

Nível formal

Informal em linguagem natural

Estilos de especificação

Nível formal

Informal em linguagem natural

Formal sintaxe e semântica formalmente definidas.

Estilos de especificação

Nível formal

Informal	em linguagem natural
Formal	sintaxe e semântica formalmente definidas.
Semi-formal	em geral apenas a sintaxe (e parte da semântica) formalmente definida.

Fatorial

```
algoritmo "fact"
var n, res: inteiro
inicio
  repita
   leia(n)
  ate n >= 0
  res <- 1
  enquanto n > 1 faca
    res <- n * res
    n <- n - 1
  fimenquanto
  escreva(res)
fimalgoritmo
```

Elementos

Estrutura básica

- Decisão (se .. então .. senão)
- Repetição (enquanto)
- E/S (leia/escreva)
- Atribuição (<-)
- Operadores matemáticos

* Fácil de entender/ler

Controle

Se. . . então. . . senão

- Execução condicional de comando(s)
- Aninhamento possível
- Condições complexas

Enquanto. . . faça

- Execução repetida de comando(s)
- Condição de continuidade
- Avaliada antes de cada execução do corpo

Construções básicas

E/S

- `leia` lê um valor da entrada padrão
- `escreva` lê um valor da saída padrão

Operações matemáticas, atribuição

- Definir um variável através um valor
- Conjunto básico de operadores

Exemplo: aprovação aluno

```
algoritmo
var n1, n2, n3, media: real
    ap: caractere
inicio
        leia(n1)
        leia(n2)
        leia(n3)
        media <- (n1 + n2 + n3) / 3
        se media < 3 entao ap <- "reprovado"
        senao se media < 5 entao ap <- "reposi"
        senao se media < 7 e n1 > 3 e n2 > 3 e n3 > 3 entao
              ap <- "aprovado"
              senao se media > 7 entao ap <- "aprovado"
                    senao ap <- "reposi"
                    fimse
              fimse
        fimse
        fimse
        escreva(ap)
fimalgoritmo
```

Fluxograma / CFG

- Uma especificação semi-formal e operacional

Fluxograma / CFG

- Uma especificação semi-formal e operacional
- Rotina: coleção de decisões manipulando, dados

Fluxograma / CFG

- Uma especificação semi-formal e operacional
- Rotina: coleção de decisões manipulando, dados
- Dados não são representados

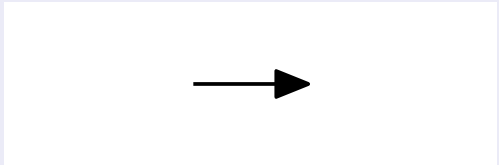
Fluxograma / CFG

- Uma especificação semi-formal e operacional
- Rotina: coleção de decisões manipulando, dados
- Dados não são representados
- Notação gráfica

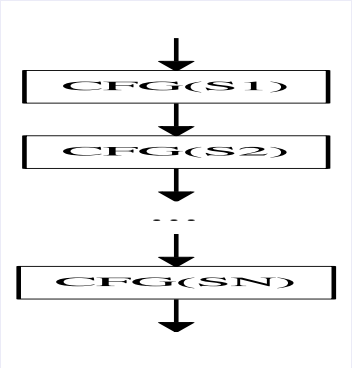
Notação gráfica

Escolha

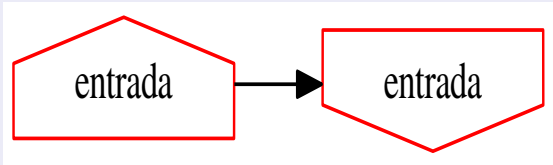
Fluxo de execução



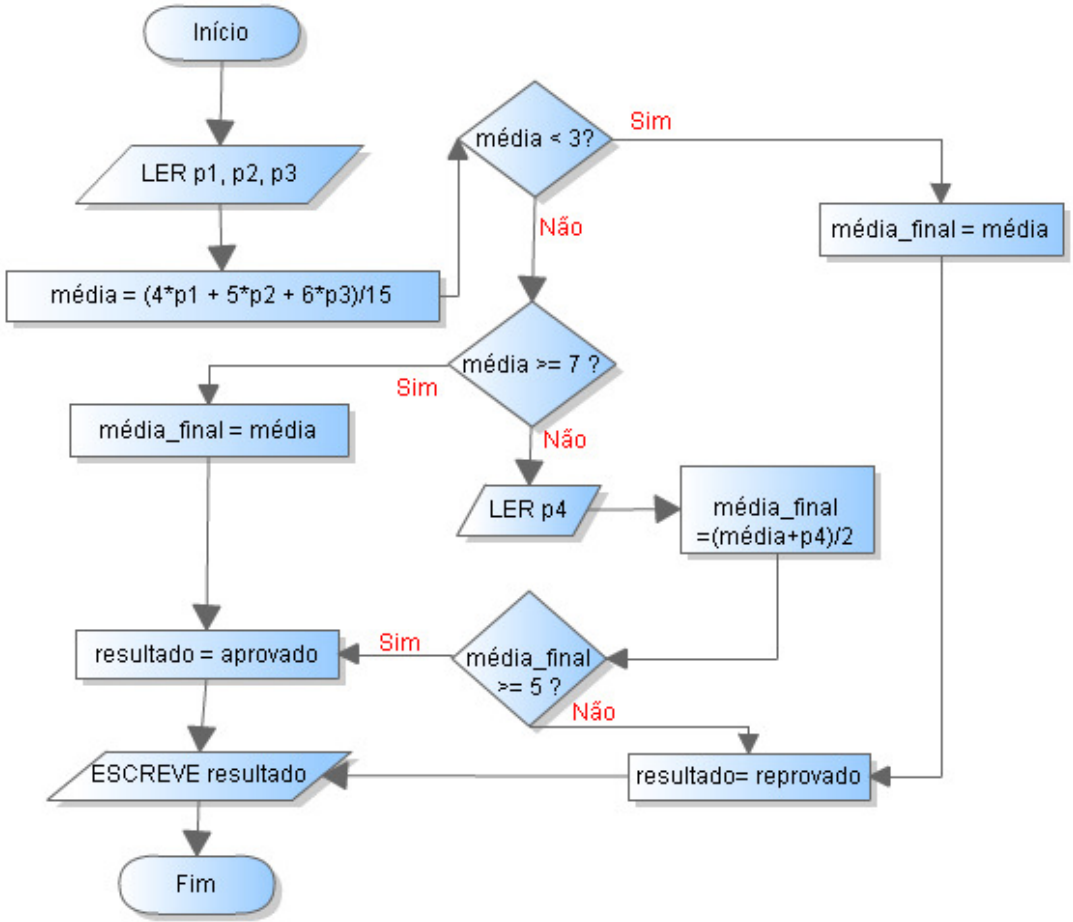
Bloco/processo



Símbolos de E/S

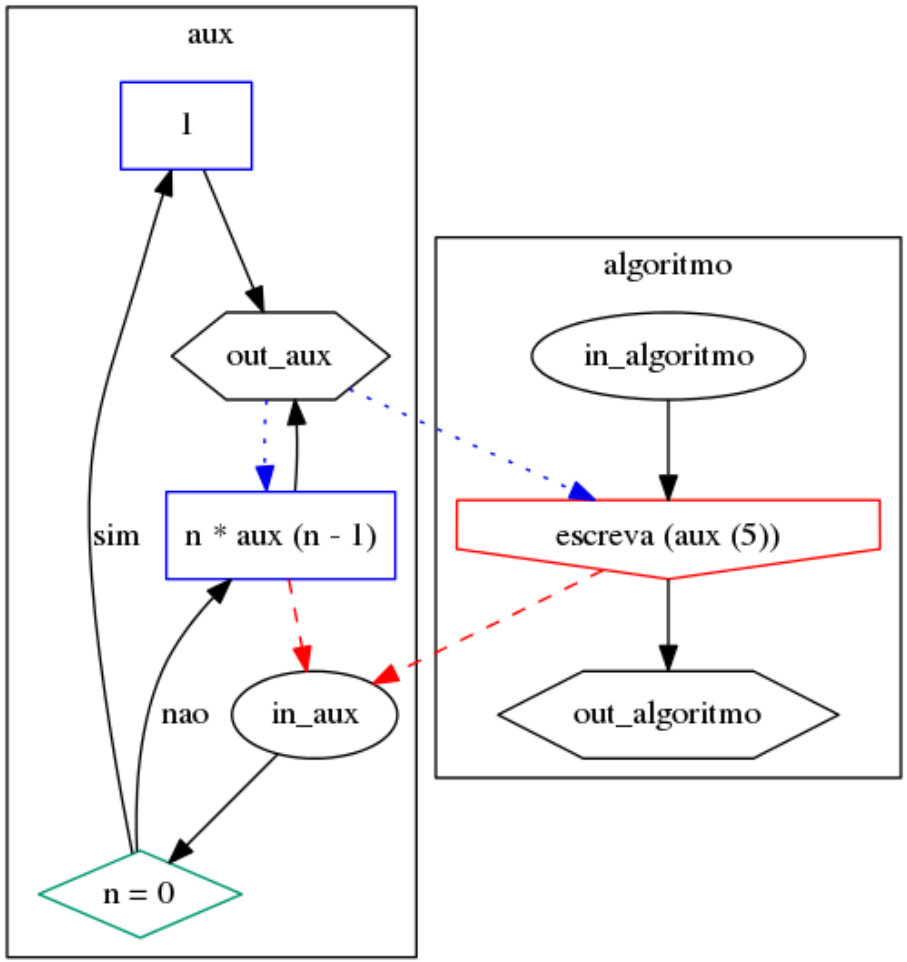


Aprovação na UFRN



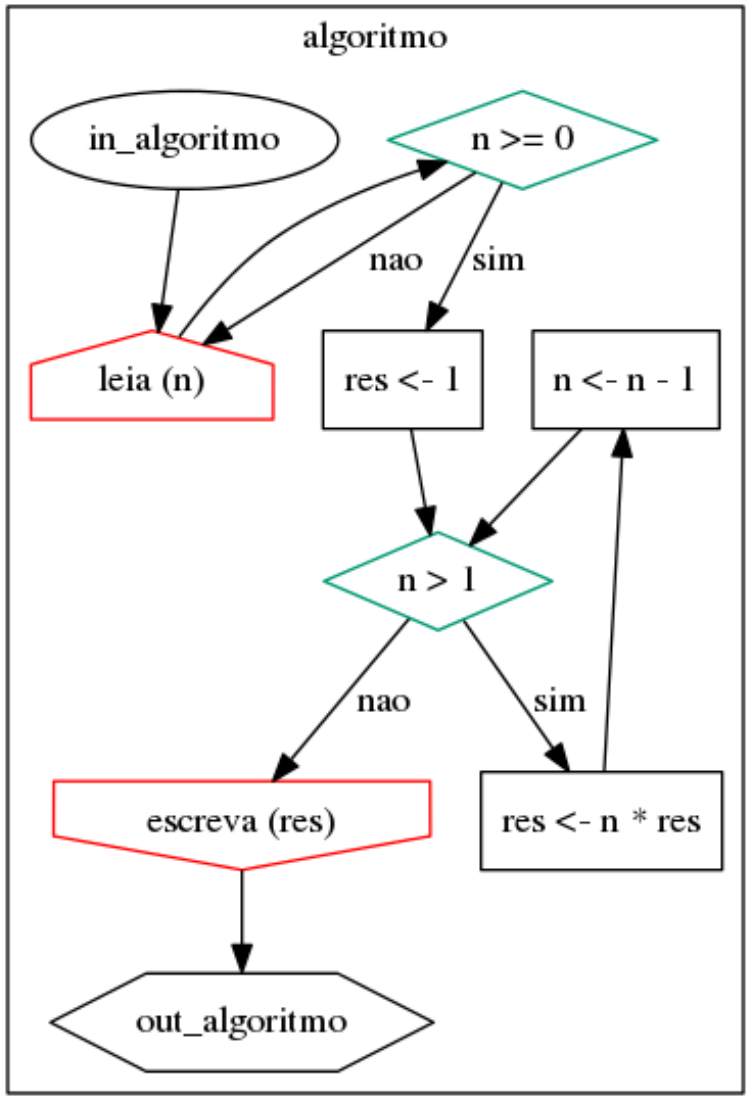
Fatorial (recursiva)

```
1   algoritmo "fact"
2   funcao aux (n: inteiro) : inteiro
3   inicio
4           se n = 0 entao retorne 1
5           senao retorne (n * (aux (n -1 )))
6           fimse
7   fimfuncao
8
9   inicio
10          escreva(aux(5))
11  fimalgoritmo
```



Fatorial (imperativa)

```
1   algoritmo "fact"
2   var n, res: inteiro
3   inicio
4     repita
5      leia(n)
6     ate n >= 0
7     res <- 1
8     enquanto n > 1 faca
9       res <- n * res
10      n <- n - 1
11    fimenquanto
12    escreva(res)
13  fimalgoritmo
```



Observações

Arestas de saída

- Arestas múltiplas

Observações

Arestas de saída

- Arestas múltiplas
- O bloco seguinte pode ser um dos sucessores na execução

Arestas de entrada

Observações

Arestas de saída

- Arestas múltiplas
- O bloco seguinte pode ser um dos sucessores na execução
- Aresta de saída = fluxo de controle saindo numa execução possível do programa

Arestas de entrada

Observações

Arestas de saída

- Arestas múltiplas
- O bloco seguinte pode ser um dos sucessores na execução
- Aresta de saída = fluxo de controle saindo numa execução possível do programa

Arestas de entrada

- Arestas múltiplas

Observações

Arestas de saída

- Arestas múltiplas
- O bloco seguinte pode ser um dos sucessores na execução
- Aresta de saída = fluxo de controle saindo numa execução possível do programa

Arestas de entrada

- Arestas múltiplas
- O controle pode vir de um dos predecessores na execução

Observações

Arestas de saída

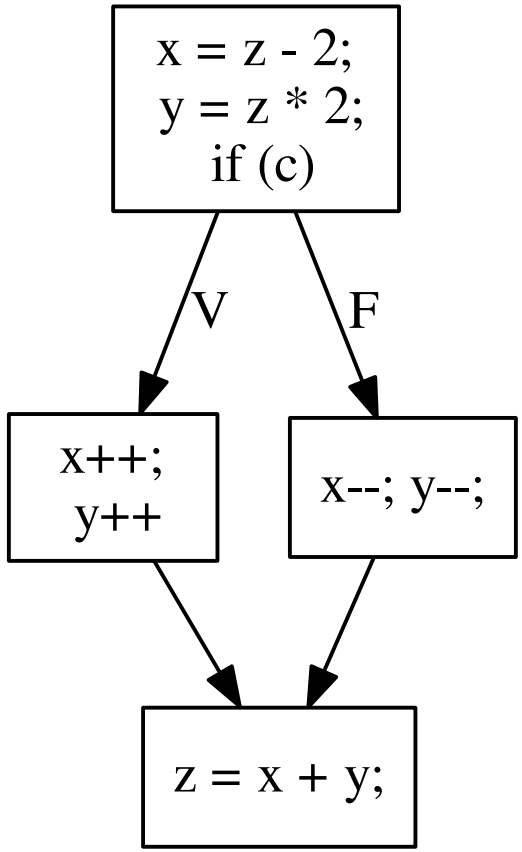
- Arestas múltiplas
- O bloco seguinte pode ser um dos sucessores na execução
- Aresta de saída = fluxo de controle saindo numa execução possível do programa

Arestas de entrada

- Arestas múltiplas
- O controle pode vir de um dos predecessores na execução
- Aresta de entrada = fluxo de controle entrando numa execução possível do programa

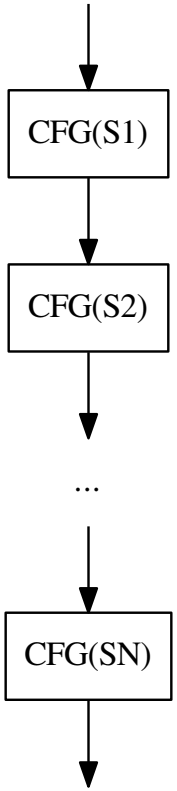
Exemplo: blocos básicos

```
1   x = z - 2;
2   y = z * 2;
3   if (c) {
4     x++;
5     y++;
6   }
7   else {
8     x--;
9     y--;
10  }
11  z = x + y;
```



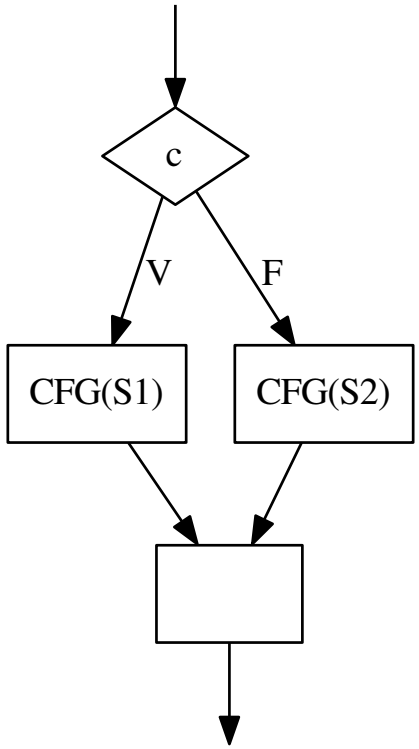
CFG bloco

- `CFG(S1; S2; ..., SN;) =`



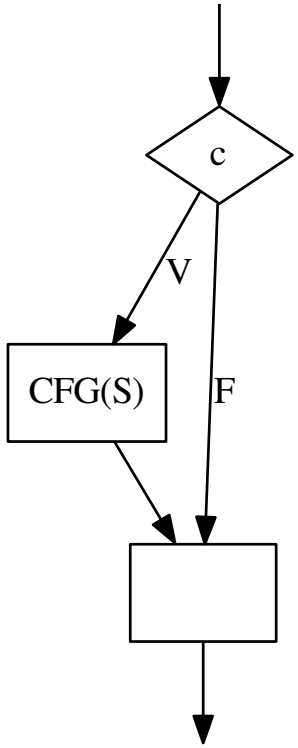
CFG if-then-else

- `CFG(if(c) then S1 else S2) =`



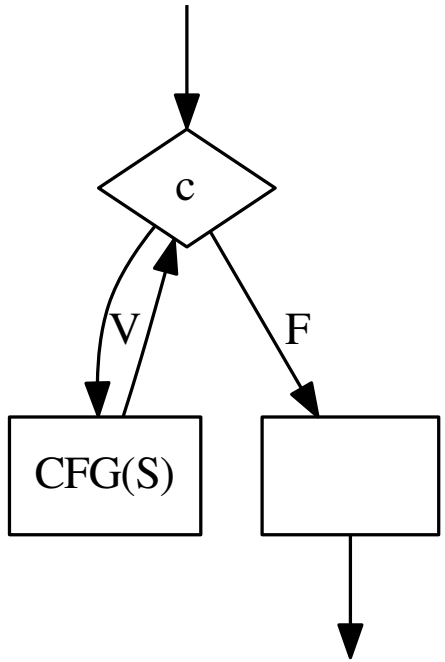
CFG if-then

- `CFG(if(c) then S) =`



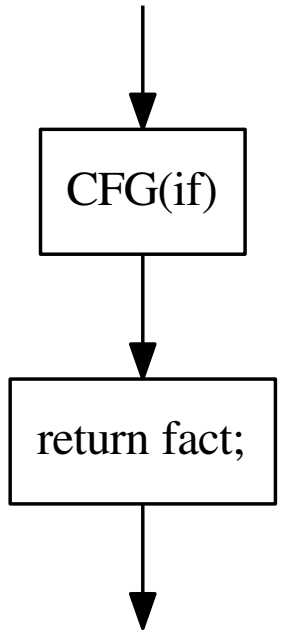
CFG while

- CFG(while(c) S) =



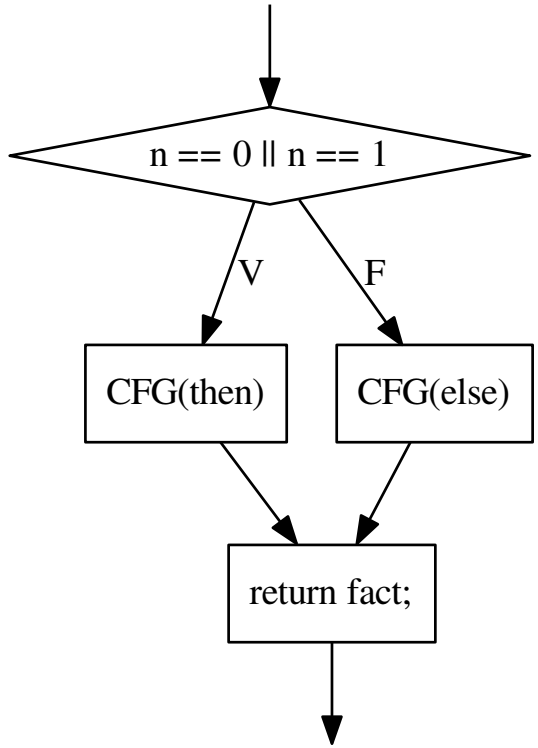
Construção recursiva

```
1   if (n == 0 || n == 1) { fact = 1; }
2    else {
3      fact = 1;
4      while (n > 1) {
5        fact = n * fact;
6        n--;
7      }
8    }
9   return fact;
```



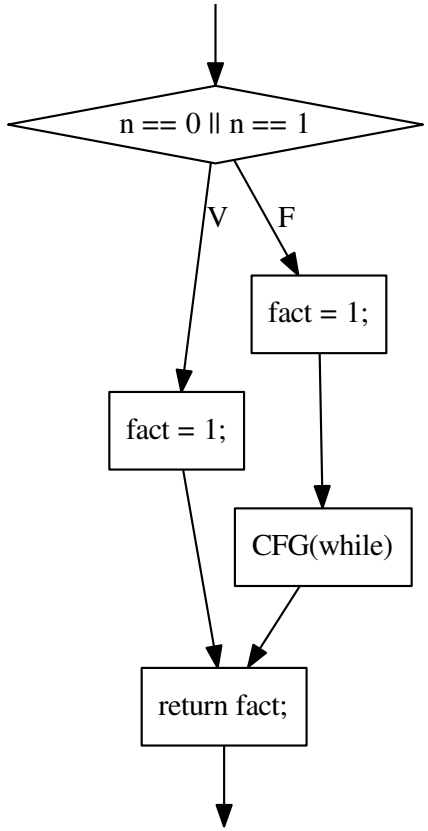
Construção recursiva

```
1   if (n == 0 || n == 1) { fact = 1; }
2    else {
3      fact = 1;
4      while (n > 1) {
5        fact = n * fact;
6        n--;
7      }
8    }
9   return fact;
```



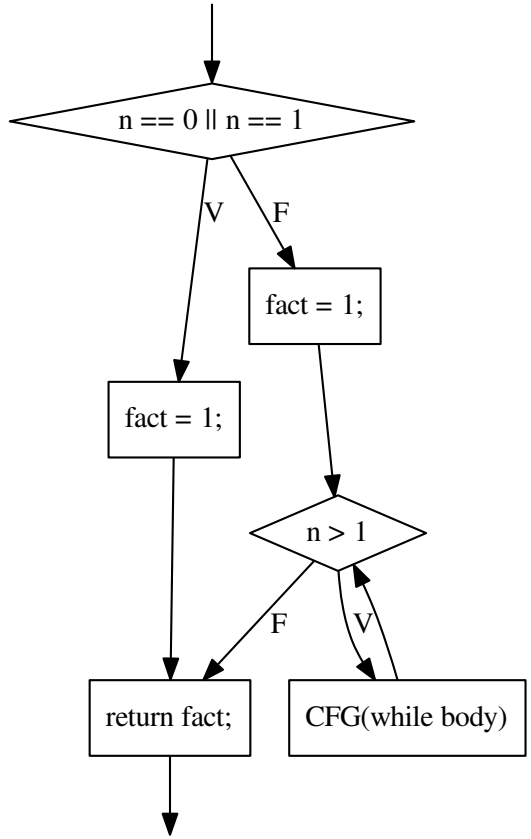
Construção recursiva

```
1   if (n == 0 || n == 1) { fact = 1; }
2    else {
3      fact = 1;
4      while (n > 1) {
5        fact = n * fact;
6        n--;
7      }
8    }
9   return fact;
```



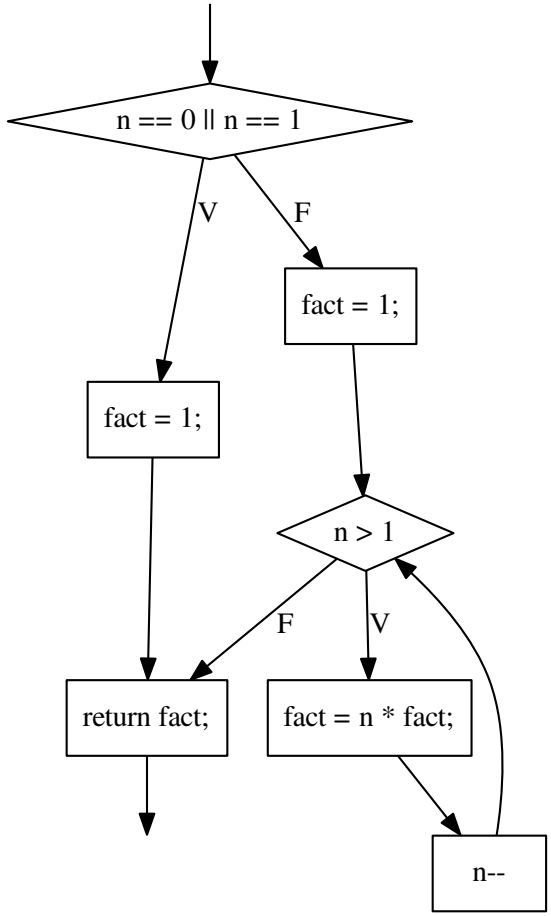
Construção recursiva

```
1   if (n == 0 || n == 1) { fact = 1; }
2    else {
3      fact = 1;
4      while (n > 1) {
5        fact = n * fact;
6        n--;
7      }
8    }
9   return fact;
```



Construção recursiva

```
1   if (n == 0 || n == 1) { fact = 1; }
2    else {
3      fact = 1;
4      while (n > 1) {
5        fact = n * fact;
6        n--;
7      }
8    }
9   return fact;
```



Uso

- O CFG é usado como representação do programa
- Assim é um quadro básico para analise estática de software (cf. interpretação abstrata e verificação dedutiva).

Exemplo de analises

- Código morte

Uso

- O CFG é usado como representação do programa
- Assim é um quadro básico para analise estática de software (cf. interpretação abstrata e verificação dedutiva).

Exemplo de analises

- Código morte
- Analise de longevidade (requisito para alocar registradores)

Uso

- O CFG é usado como representação do programa
- Assim é um quadro básico para analise estática de software (cf. interpretação abstrata e verificação dedutiva).

Exemplo de analises

- Código morte
- Analise de longevidade (requisito para alocar registradores)
- Qualquer execução simbólica

Do fluxograma para o código

- Pode fazer uma especificação da tradução inversa ?
- Por exemplo de fluxogramas para um C básico ?

DFD

- Uma especificação semi-formal e operacional

DFD

- Uma especificação semi-formal e operacional
- Sistema = coleção de dados manipulados por processos

DFD

- Uma especificação semi-formal e operacional
- Sistema = coleção de dados manipulados por processos
- Dados:

DFD

- Uma especificação semi-formal e operacional
- Sistema = coleção de dados manipulados por processos
- Dados:
  - podem ser persistentes (repositórios de dados)

DFD

- Uma especificação semi-formal e operacional
- Sistema = coleção de dados manipulados por processos
- Dados:
  - podem ser persistentes (repositórios de dados)
  - podem fluir (fluxo de dados)

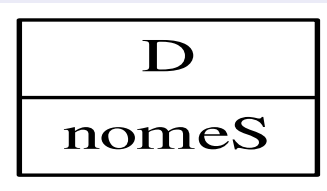
DFD

- Uma especificação semi-formal e operacional
- Sistema = coleção de dados manipulados por processos
- Dados:
  - podem ser persistentes (repositórios de dados)
  - podem fluir (fluxo de dados)
- Notação gráfica

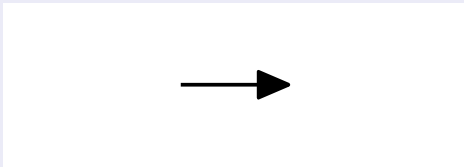
Notação gráfica

Processo

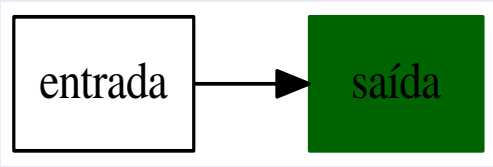
Depósito



Fluxo de dado



Terminado (E/S)



Processo

- Transformação $E \rightarrow S$
- Nome (verbo + objeto)
- Não deve expor nada da implementação
- Numerar os processos para diagramas maiores

Fluxo

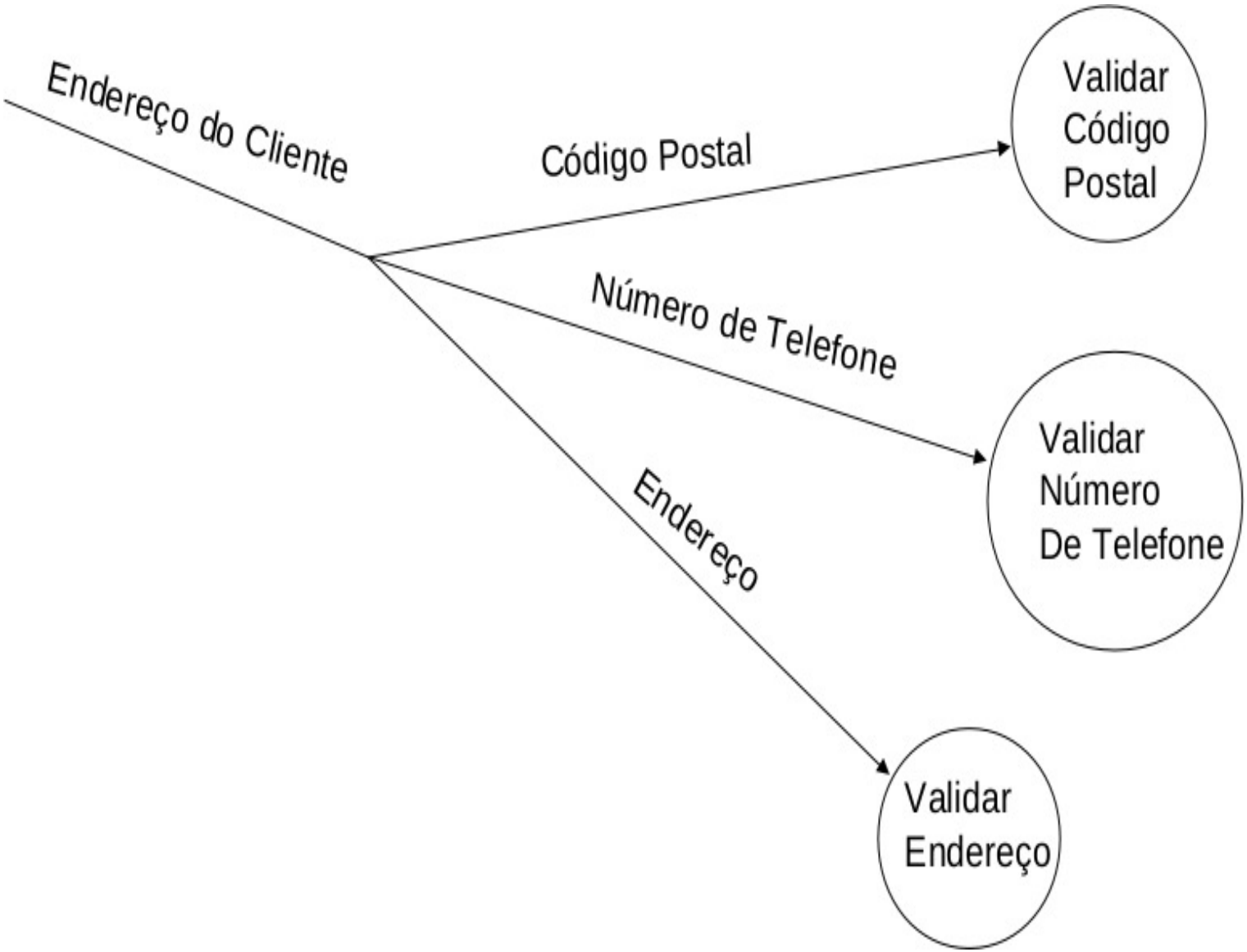
Représentation

- Dados em movimento
- Nomeado ou não
- Caso sim, nome = dados/tipo de dados em fluxo
- Direção do fluxo de dados

Características

- Fluxos convergentes
  - Dados agregados para formar um fluxo maior
  - Nomeados
- Fluxos divergentes
  - Divisão em parte mais simples
  - Mesmo dado para entidades diferentes

Exemplo de fluxo



Depósito

- Conjunto de dados em repouso

Depósito

- Conjunto de dados em repouso
- Nome no plural

Depósito

- Conjunto de dados em repouso
- Nome no plural
- Um depósito não altera seus próprios dados

Depósito

- Conjunto de dados em repouso
- Nome no plural
- Um depósito não altera seus próprios dados
- É geralmente um banco de dados (pode ser discos, backup, ..)

Depósito

- Conjunto de dados em repouso
- Nome no plural
- Um depósito não altera seus próprios dados
- É geralmente um banco de dados (pode ser discos, backup, ..)
- Os fluxos chegando:

Depósito

- Conjunto de dados em repouso
- Nome no plural
- Um depósito não altera seus próprios dados
- É geralmente um banco de dados (pode ser discos, backup, ..)
- Os fluxos chegando:
  - Traduzem pedidos de inclusão, alteração, exclusão

Depósito

- Conjunto de dados em repouso
- Nome no plural
- Um depósito não altera seus próprios dados
- É geralmente um banco de dados (pode ser discos, backup, ..)
- Os fluxos chegando:
  - Traduzem pedidos de inclusão, alteração, exclusão
  - Devem transportar dados do tipo adequado

Depósito

- Conjunto de dados em repouso
- Nome no plural
- Um depósito não altera seus próprios dados
- É geralmente um banco de dados (pode ser discos, backup, ..)
- Os fluxos chegando:
  - Traduzem pedidos de inclusão, alteração, exclusão
  - Devem transportar dados do tipo adequado
- Os fluxos saindo:

Depósito

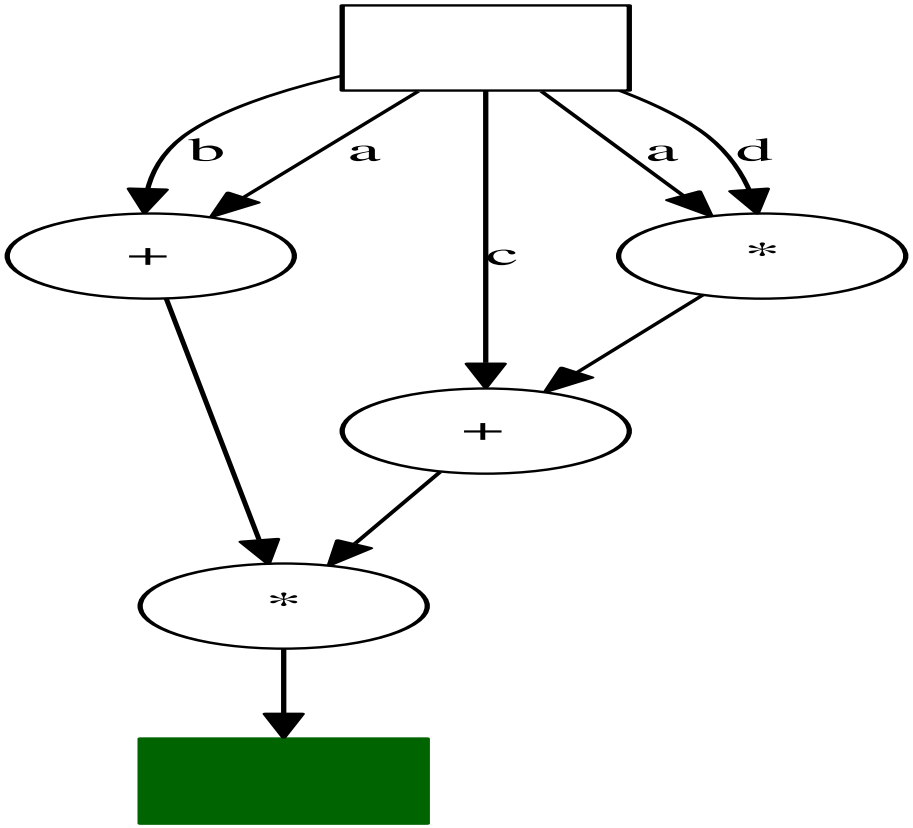
- Conjunto de dados em repouso
- Nome no plural
- Um depósito não altera seus próprios dados
- É geralmente um banco de dados (pode ser discos, backup, ..)
- Os fluxos chegando:
  - Traduzem pedidos de inclusão, alteração, exclusão
  - Devem transportar dados do tipo adequado
- Os fluxos saindo:
  - Leitura de dados (um, alguns, todos)

Terminador

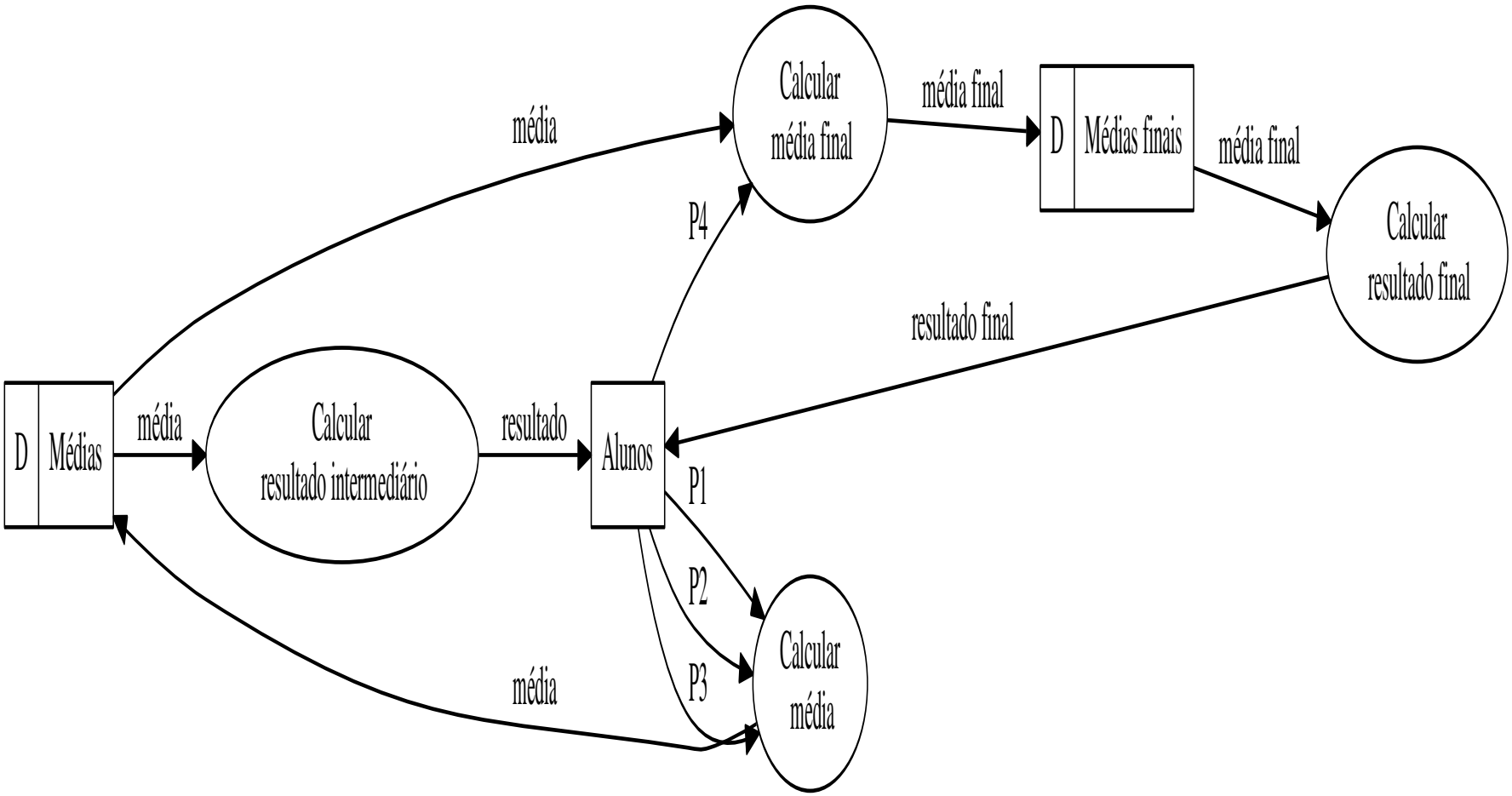
- Entidade externa que comunica com o programa
- Interface do programa

Exemplo: expressão aritmética

- $(a + b) * (c + a * d)$

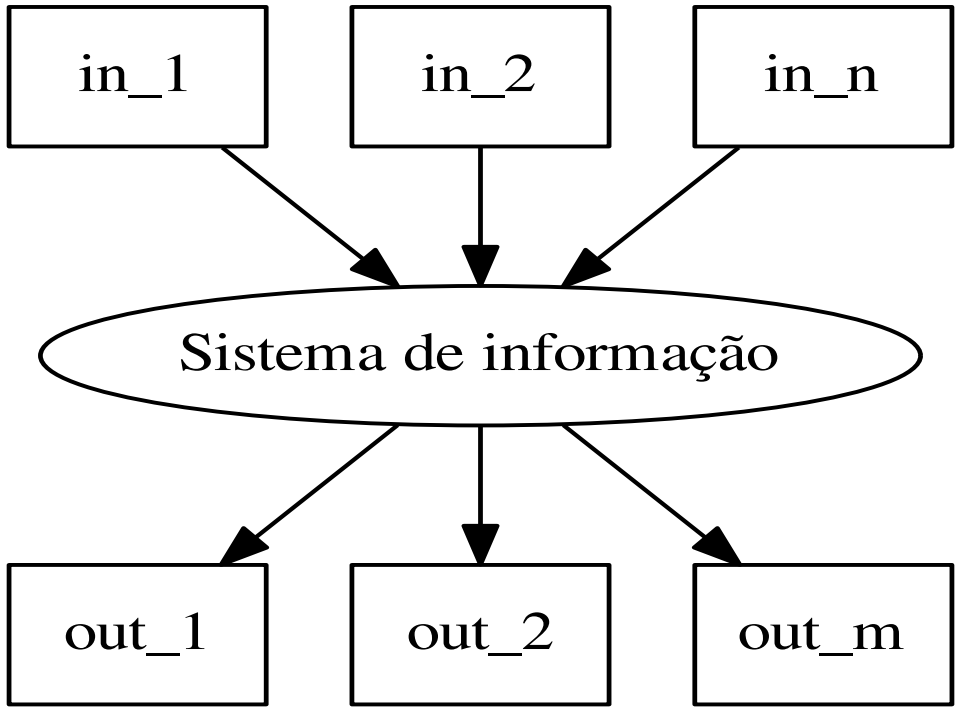


Aprovação na UFRN



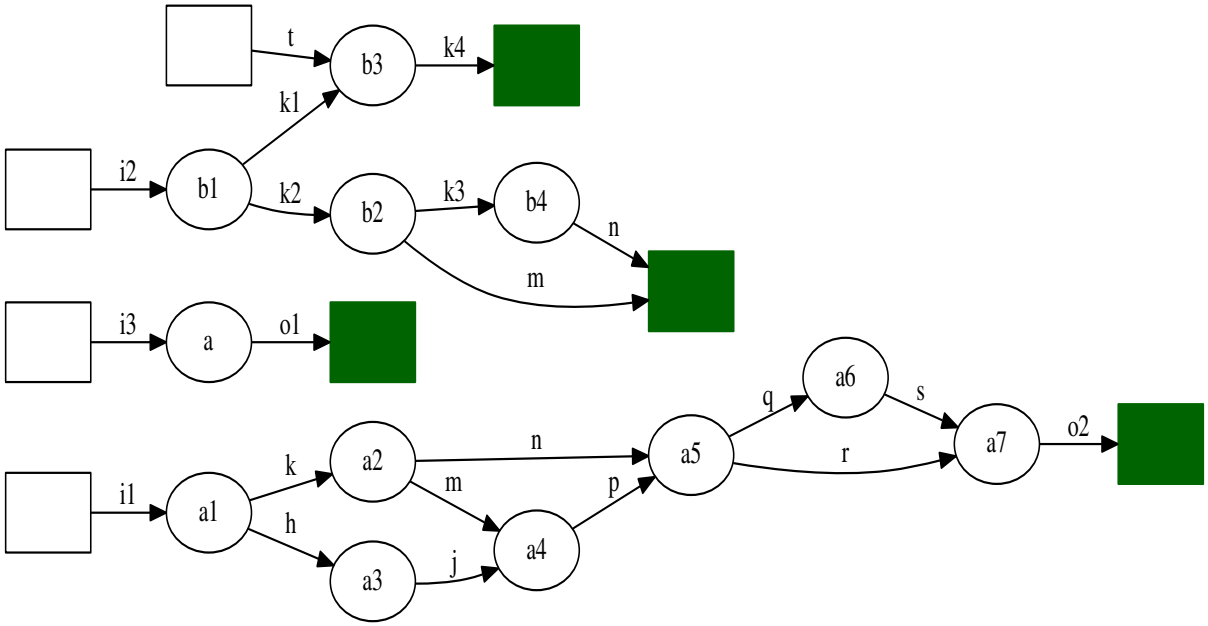
Construção (1)

- Usar o diagrama de "contexto"

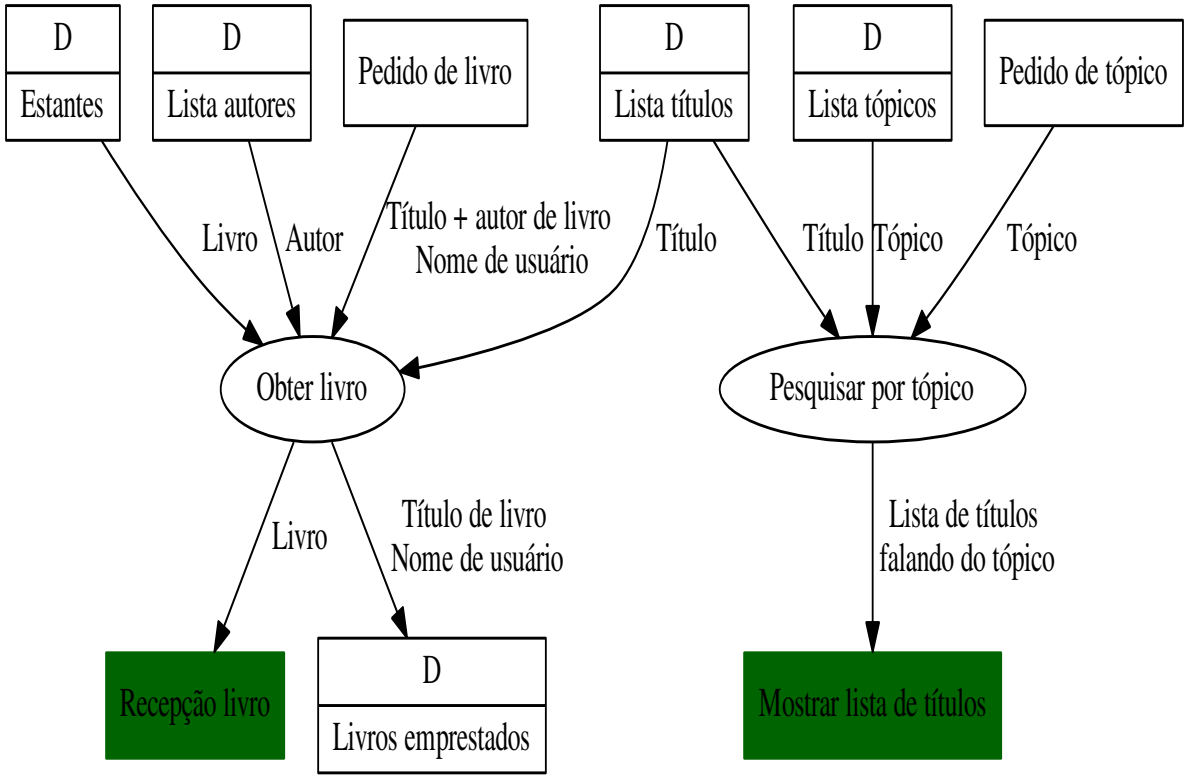


Construção (2)

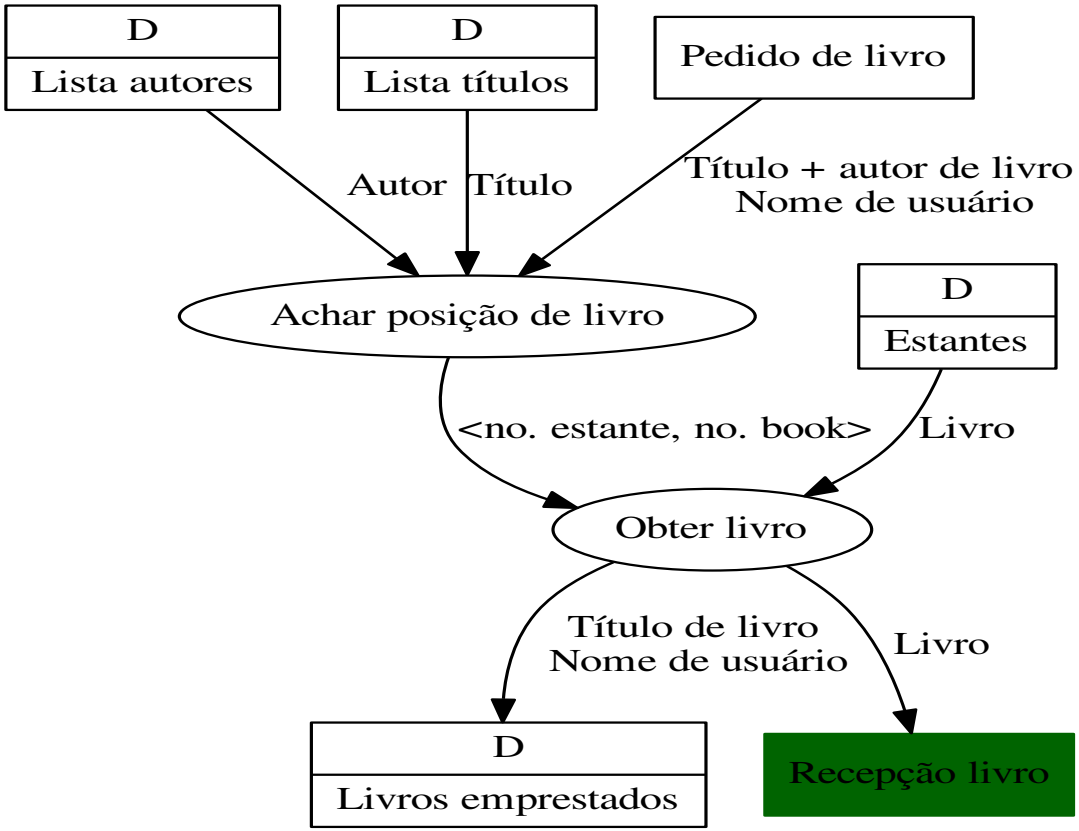
- Refinar ate chegar às funções elementárias
  - equilibrar os refinamentos



Uma biblioteca



Refinamento



Resumo

Perguntas ?

http://dimap.ufrn.br/~richard/dim0436