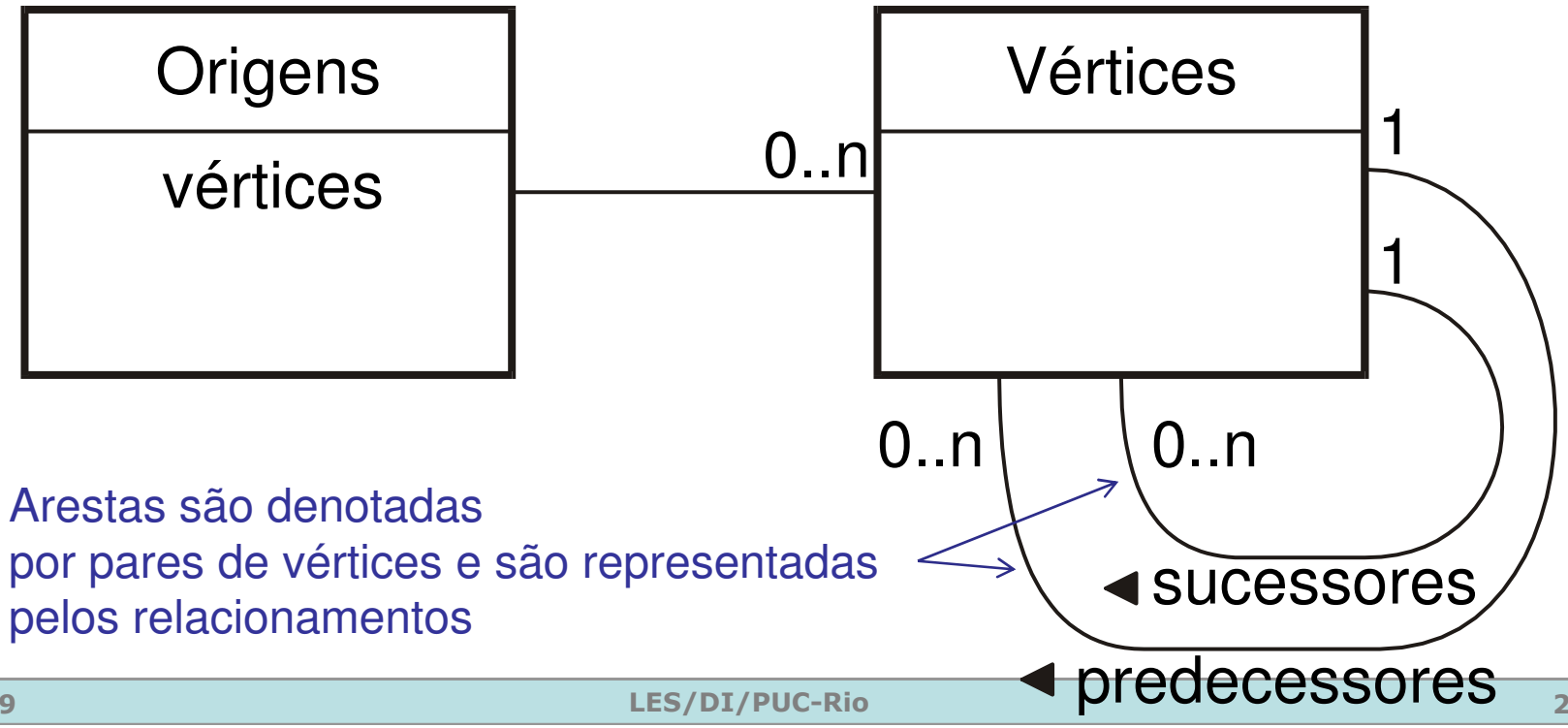
Aula 15

Modelagem Física

Alessandro Garcia
LES/DI/PUC-Rio
Setembro 2010

Relembrando... O que é um modelo?

- Um modelo é uma abstração a partir da qual podem-se avaliar, de forma racional, as propriedades de um conceito
- Qualquer modelo deve descrever o conjunto de todas as instâncias possíveis do conceito modelado
- **Modelo conceitual de um grafo dirigido**:



Modelos vs. exemplos

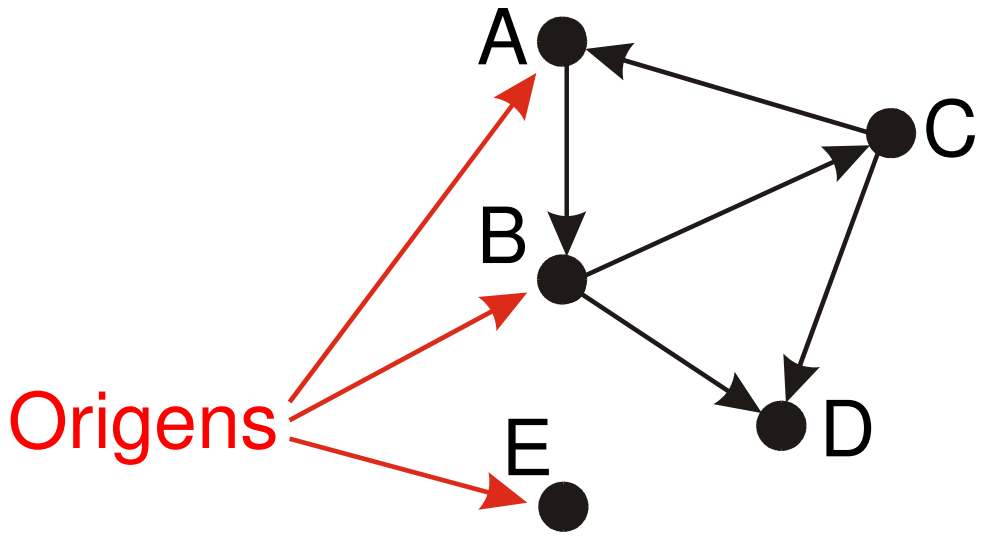
- Uma instância do modelo corresponde a um **exemplo**
- Cada exemplo precisa **satisfazer as propriedades estabelecidas** pelo modelo e suas assertivas estruturais
- Objetivos dos exemplos:
  - Ilustrar uma ou mais instâncias de estruturas que podem ser obtidas a partir do modelo
  - Exemplos concretos são mais fáceis de entender do que puramente ler modelos abstratos
  - Podem ajudar a apontar casos válidos que não são permitidos pelo modelo
- Entretanto, *exemplos não podem servir como especificação*
  - somente complementam
  - um conjunto de exemplos dificilmente captura todas as características de estruturas de dados

A importância de modelos visuais

- Um exemplo conceitual é uma das possíveis instâncias de um modelo conceitual
  - dito de outra forma, um exemplo conceitual é um conjunto de elementos concretos que satisfaz o modelo conceitual
- Exemplo (instância) de um grafo
  - $\mathcal{V}$ = { A , B , C , D , E }
  - $\mathcal{A}$ = { <A,B> , <B,C> , <B,D> , <C,A> , <C,D> }
  - $\mathcal{O}$ = { A , B , E }
- Entenderam o exemplo?
  - o grafo é conexo?
  - a partir das origens podem-se alcançar todos os vértices do grafo?

Representação gráfica do exemplo conceitual

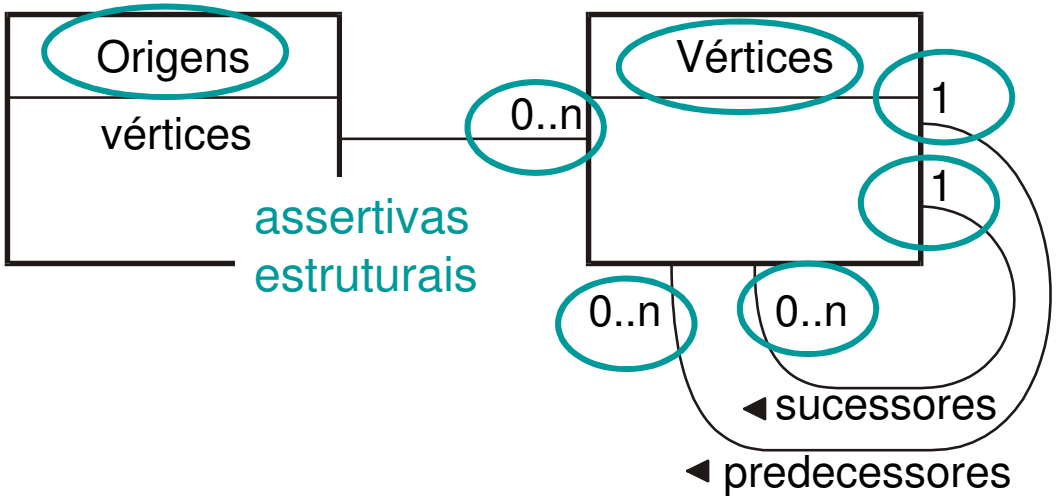
- Figuras valem por mil palavras
  - uma representação gráfica é muito melhor para a compreensão humana
    - para exemplos utilizam-se em geral representações (ad hoc) que mais nos convierem



  - em um modelo conceitual da arquitetura, também poderíamos analisar propriedades de forma similar: p.e. dependências entre módulos

Do modelo conceitual para...

- ... modelo físico:
  - É o resultado de uma **transformação do modelo conceitual**, tornando-o realizável no meio físico alvo
  - Estabelecem-se **interdependências** a serem realizadas no **meio físico escolhido**, tais como:
    - memória principal, determinada linguagem de programação, ou determinado sistema de gerência de banco de dados
  - Transformações devem **preservar** as características do modelo conceitual
  - Exemplo – **Grafo direcionado**



Esta aula: especificação

- Aula Passada: Modelo conceitual & exemplo conceitual
- Objetivo dessa aula
  - Apresentar os conceitos e notações de modelagem física
  - Apresentar uma linguagem gráfica para a modelagem física de estruturas de dados
  - Motivar o uso de assertivas com modelos
- Referência básica:
  - Capítulo 9.4
- Referência complementar
  - Silva, R.P.; *UML2 em Modelagem Orientada a Objetos*; Florianópolis, SC: Visual Books; 2007

Sumário

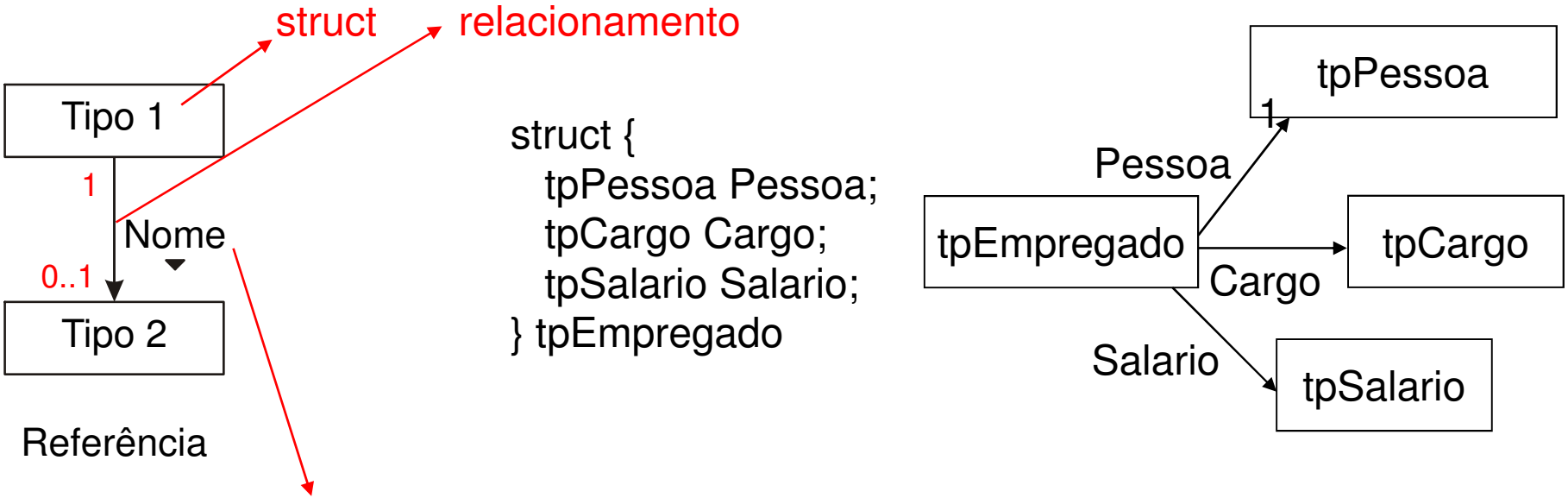
- Linguagem de representação gráfica para modelos físicos
- Modelo físico
- Exemplo físico
- Modelagem de cabeças de estrutura de dados
- Assertivas estruturais

UML (Unified Modelling Language)

- UML: foi desenvolvida para...
  - modelagem orientada à objetos: classes, objetos, e diferentes relacionamentos
  - modelagem conceitual: domínio da aplicação (“entidades do mundo real”)
- Modelagem física com UML requer ligeiras adaptações
  - utiliza-se **notação semelhante**
- Um modelo físico de uma estrutura de dados descreve todas as possíveis instâncias desse conceito em memória física
- Um exemplo físico ilustra exatamente uma dessas possíveis instâncias

Linguagem de representação

- Estruturas de dados físicas podem ser modeladas usando uma representação semelhante a dos **diagramas de classes** da UML2
  - quatro tipos de relacionamentos

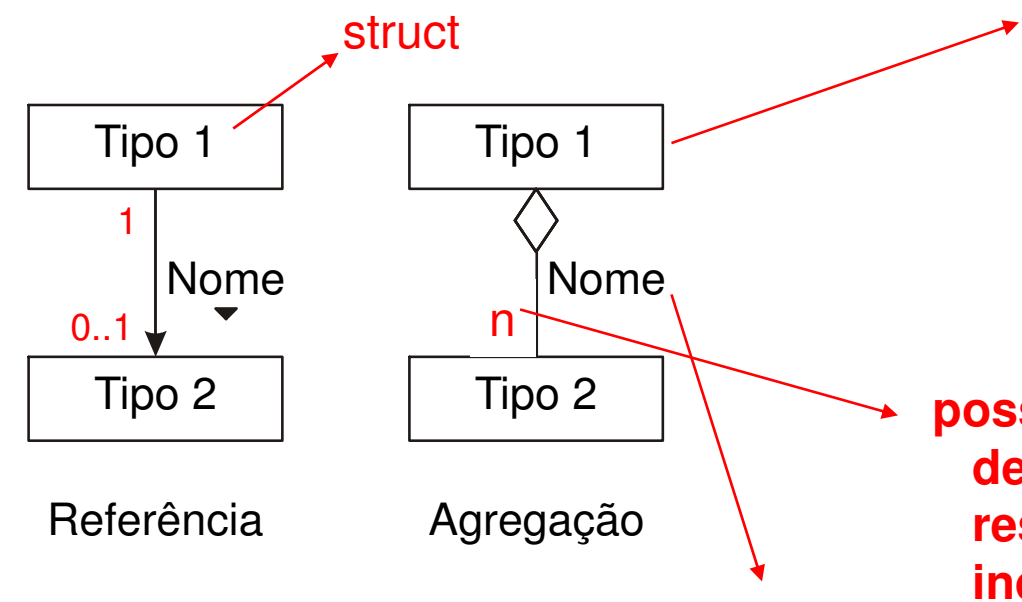


(outro exemplo: ponteiros)

- **Tipo 1 e Tipo 2** são agregados de 1 ou mais elementos de dados: ou seja, são estruturas (struct)

Linguagem de representação física

- Estruturas de dados físicas podem ser modeladas usando uma representação semelhante a dos **diagramas de classes** da UML2
  - quatro tipos de relacionamentos



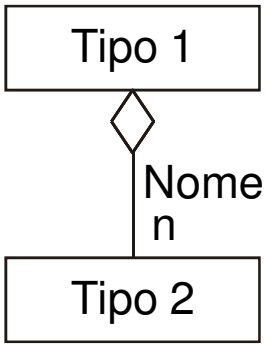
Cuidado! Destruição de elemento composto freqüentemente requer 1º. a destruição dos constituintes

possibilidades:
definido, exemplo: 20
restrito, exemplo: 0..20
indefinido, exemplos: *
+ (1 ou mais)
3..*

- **Tipo 1 e Tipo 2** são agregados de 1 ou mais elementos de dados: ou seja, são estruturas (struct)

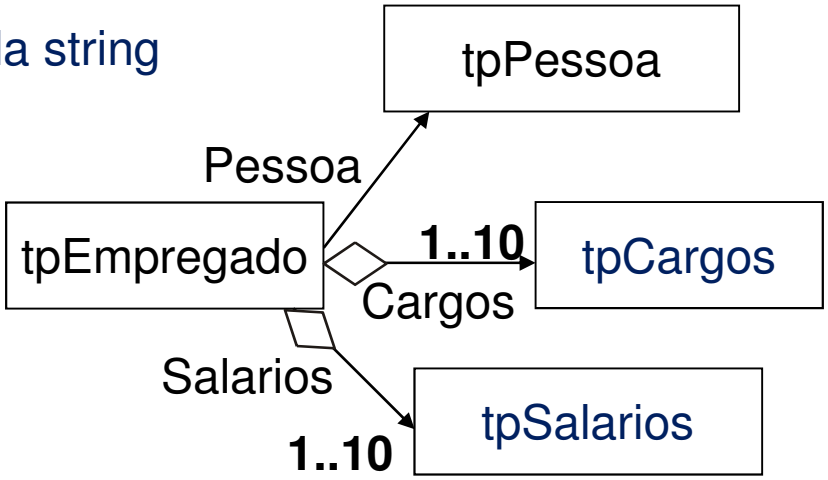
Linguagem de representação física

- Estruturas de dados físicas podem ser modeladas usando uma representação semelhante a dos **diagramas de classes** da UML2
  - quatro tipos de relacionamentos



Agregação

```
struct {
   tpPessoa Pessoa;
   char nomesCargos[10][X];
      // X é o tam. max. de cada string
   int  valoresSalarios[10];
} tpEmpregado
```

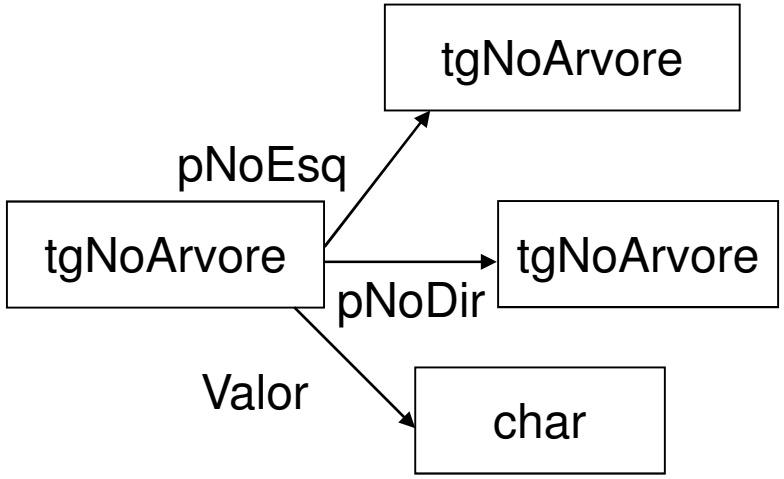


- **Tipo 1 e Tipo 2** são agregados de 1 ou mais elementos de dados: ou seja, são estruturas (struct) ou classes

Outro Exemplo

- Modelando fisicamente a estrutura de uma Arvore (ignorando a cabeça)

```
typedef struct tgNoArvore {

        struct tgNoArvore * pNoEsq ;
            /* Ponteiro para filho à esquerda

        struct tgNoArvore * pNoDir ;
            /* Ponteiro para filho à direita

        char Valor ;
            /* Valor do nó */

 } tpNoArvore ;
```

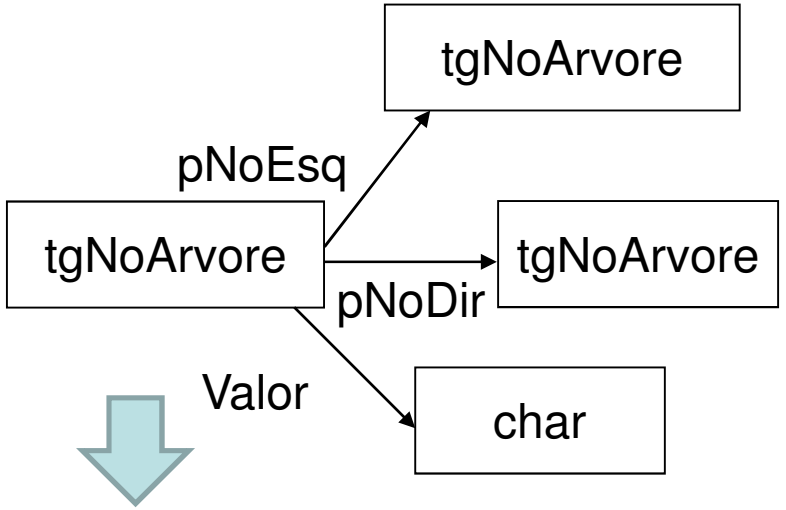


Como simplificar notação?

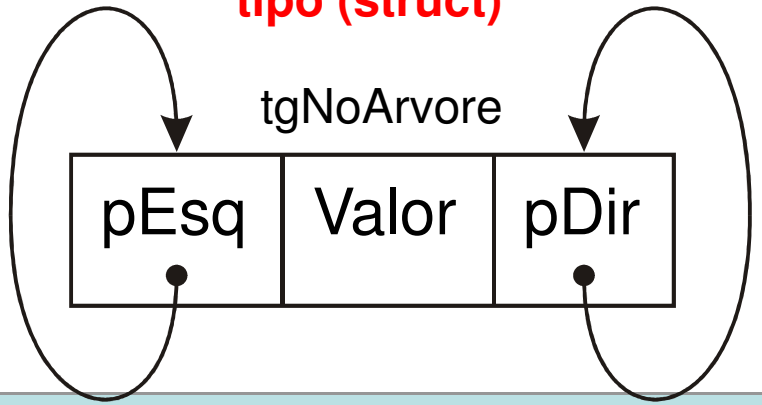
Outro Exemplo

- Modelando fisicamente a estrutura de uma Arvore (ignorando a cabeça)

```
typedef struct tgNoArvore {

        struct tgNoArvore * pNoEsq ;
            /* Ponteiro para filho à esquerda

        struct tgNoArvore * pNoDir ;
            /* Ponteiro para filho à direita

        char Valor ;
            /* Valor do nó */

 } tpNoArvore ;
```

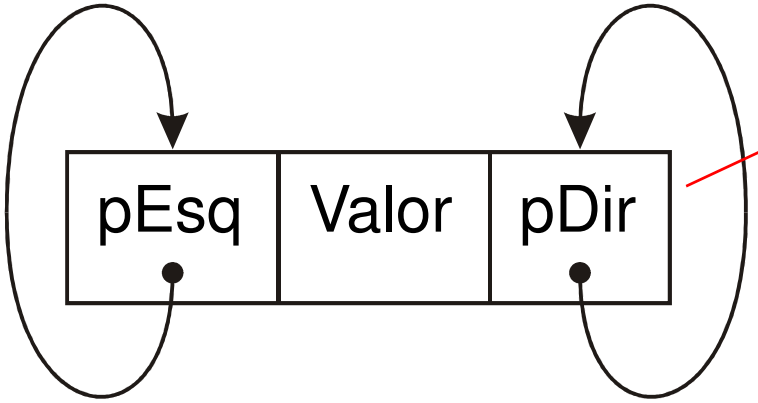


forma concisa de representar um tipo (struct)



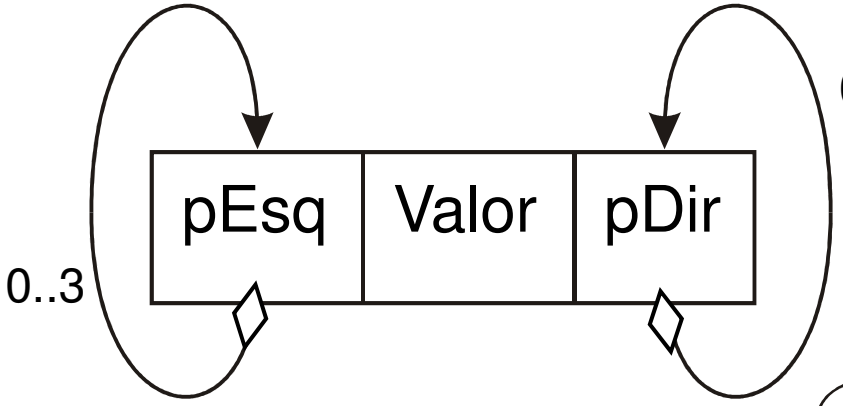
Exemplo: Comparando Referência vs. Agregação

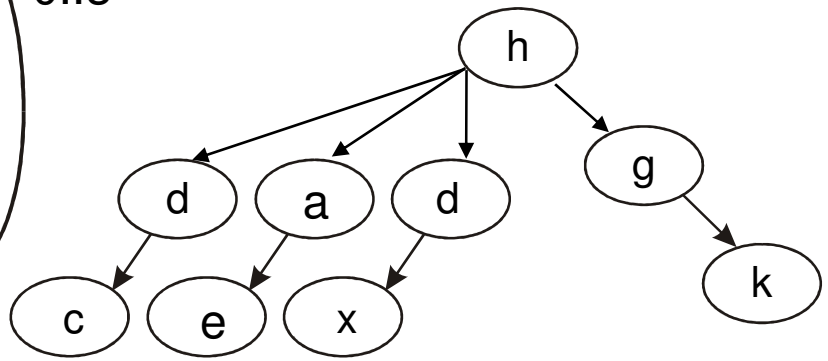
Modelo de árvore binária usando referências



forma concisa de representar um tipo (struct)

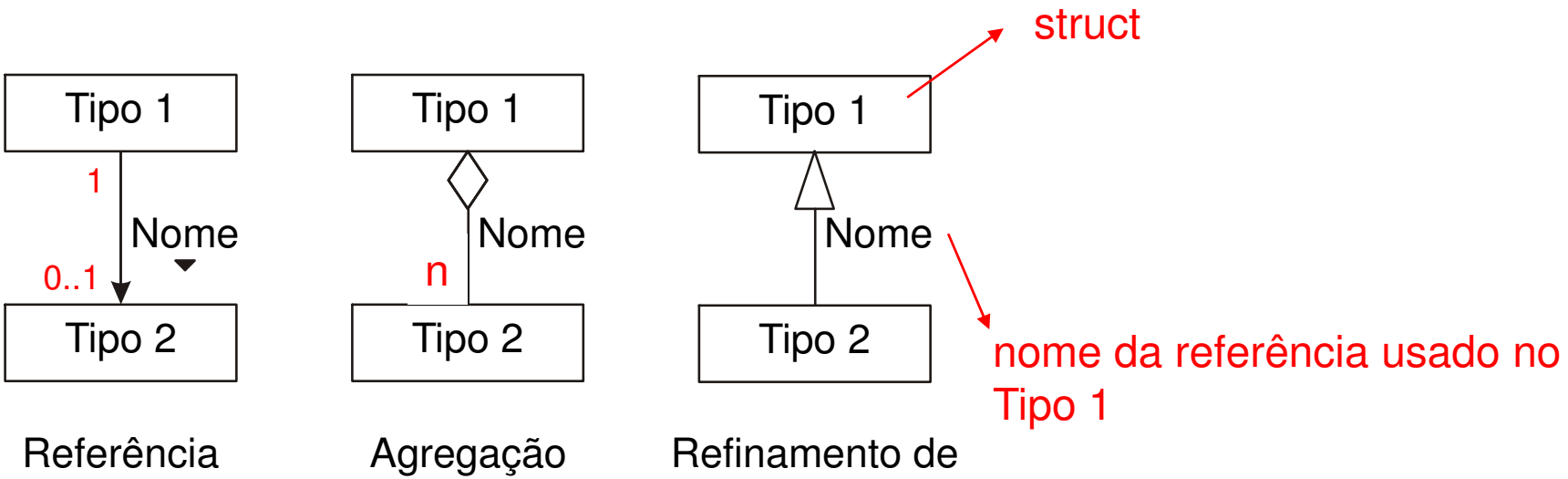
Modelo de árvore n-ária usando agregação





Linguagem de representação

- Estruturas de dados físicas podem ser modeladas usando uma representação semelhante a dos **diagramas de classes** da UML2
  - quatro tipos de relacionamentos



- **Tipo 1 e Tipo 2** são agregados de 1 ou mais elementos de dados: ou seja, são estruturas (struct) ou classes

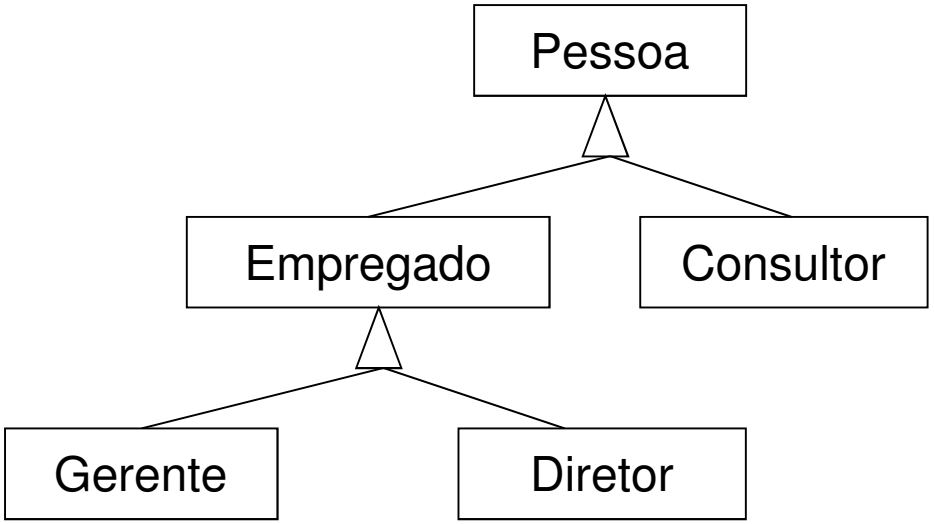
Como implementar?

- herança:
  - Java e C++: construções de herança disponíveis na linguagem
  - C: também é possível
    - conjunto de *structs*

Como implementar?

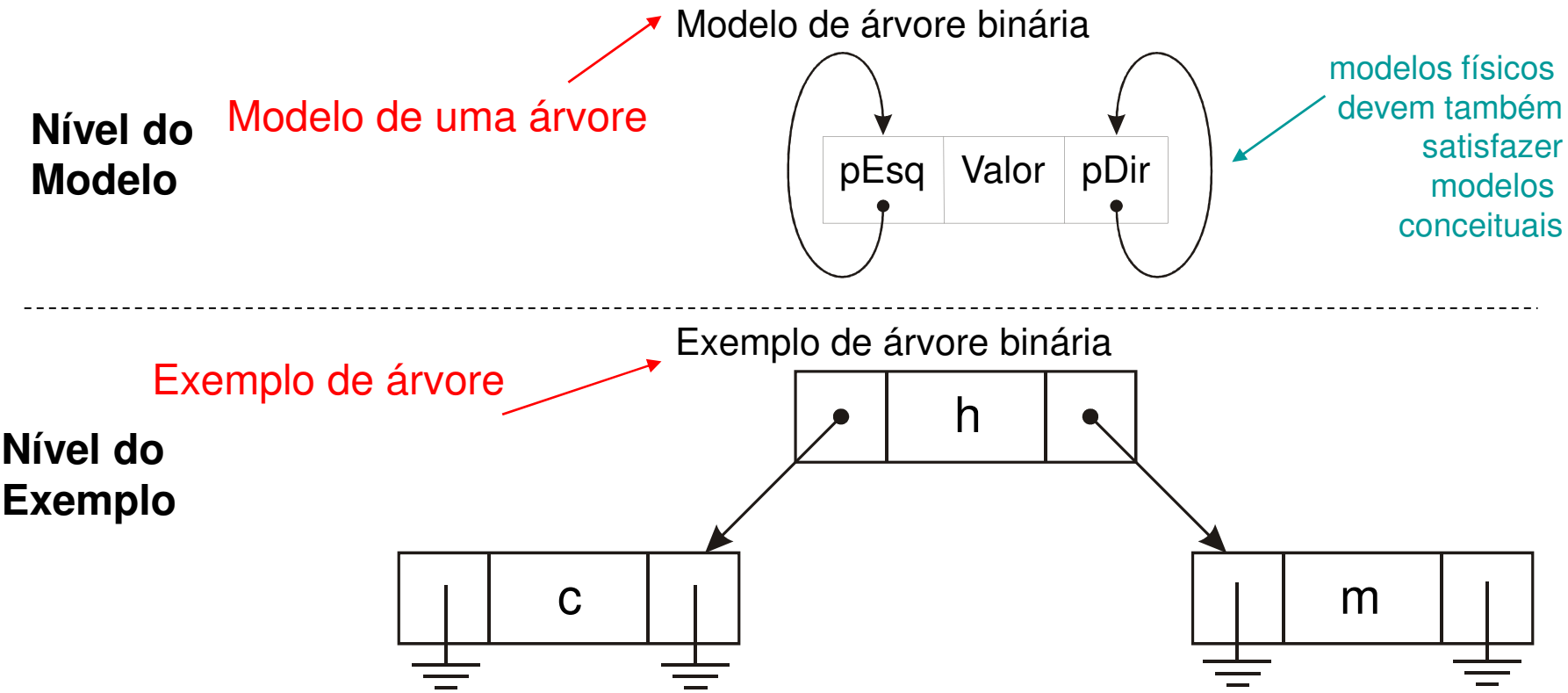
- Herança em C:

```
typedef struct {
   /* entram aqui os dados de pessoa
} tpPessoa

struct {
   tpPessoa Pessoa ;
   /* entram aqui os dados de empregado
} tpEmpregado

struct {
  tpEmpregado Empregado
   /* entram aqui os dados de gerente
} tpGerente

struct {
   tpEmpregado Empregado
   /* entram aqui os dados de diretor
} tpDiretor

…}
```

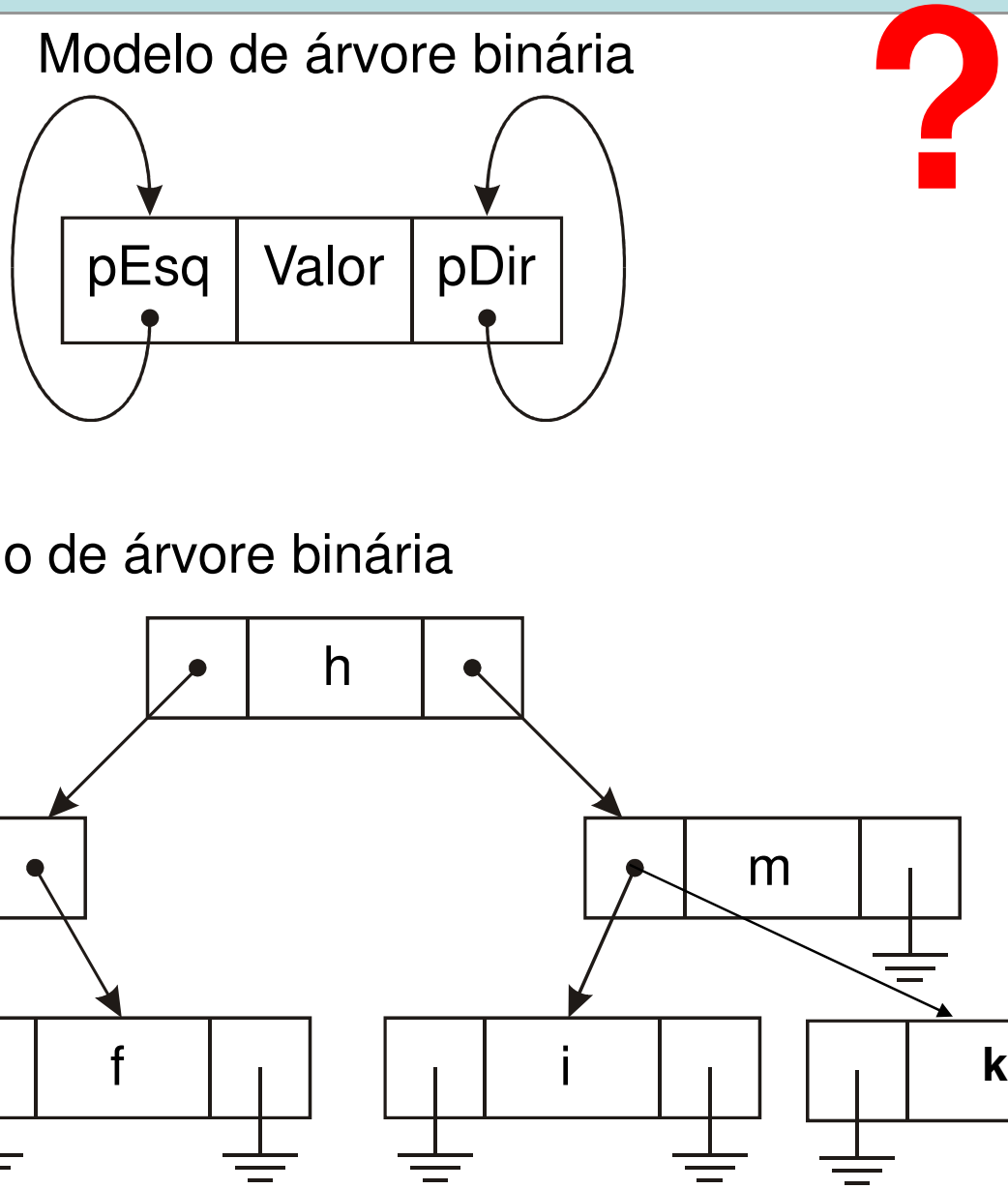


Modelo x Exemplo

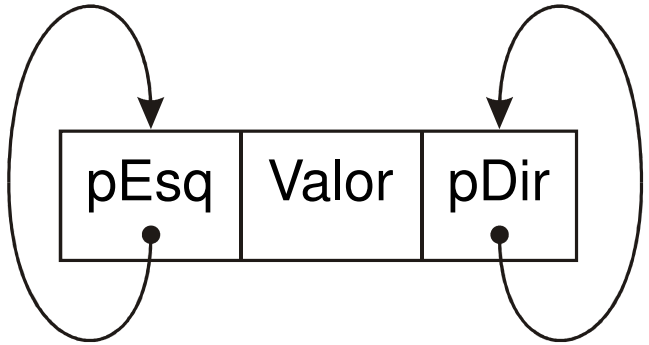
- Modelagem física em níveis:



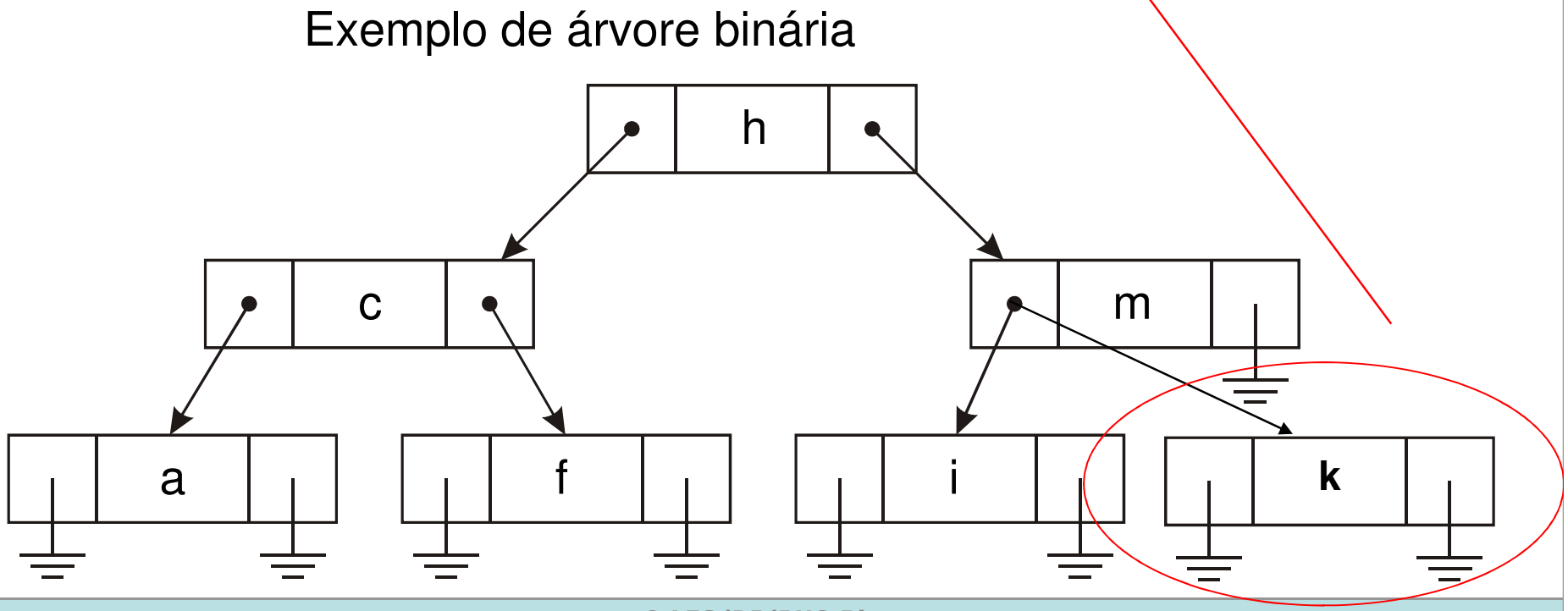
É um exemplo válido do modelo?



É um exemplo válido do modelo?



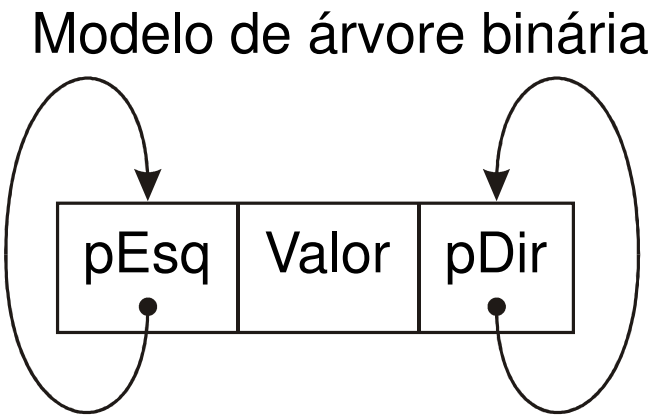
assumindo que o exemplo fosse válido, como modificar o modelo



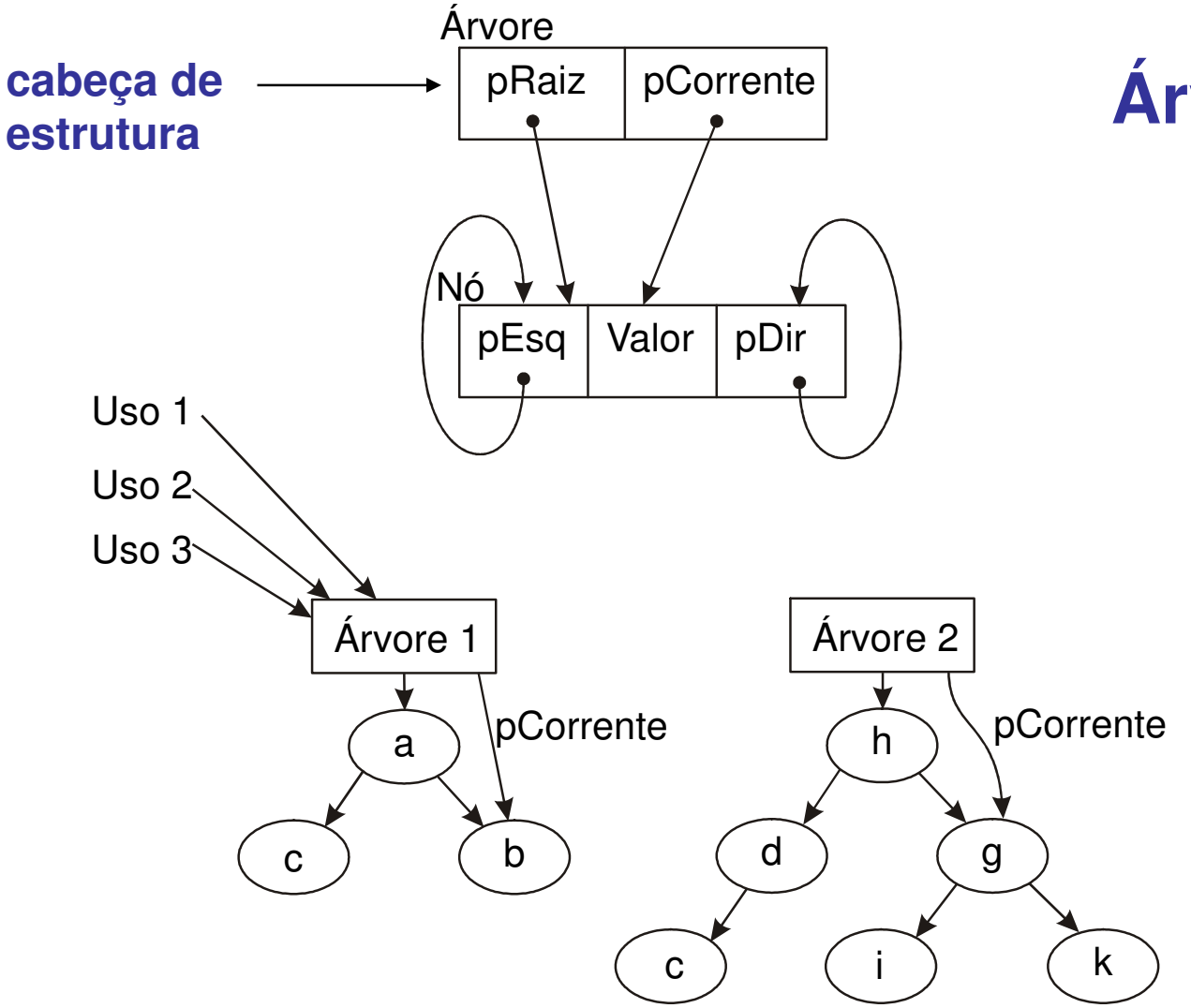
Se fossem modelar por completo...

... a estrutura de dados Árvore (Trab 1):

o que falta neste modelo?



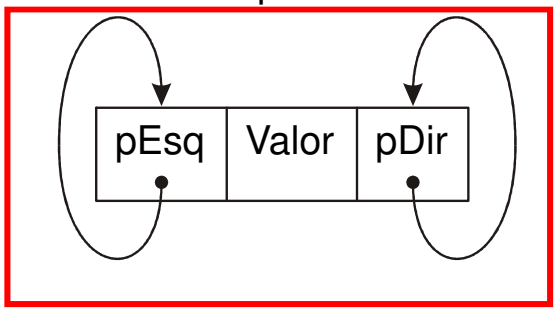
Como representar cabeças de estruturas?



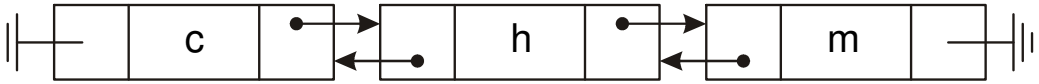
Cuidado com Figuras

Modelos demasiadamente abstratos

Modelo de lista duplamente encadeada



Exemplo de lista duplamente encadeada

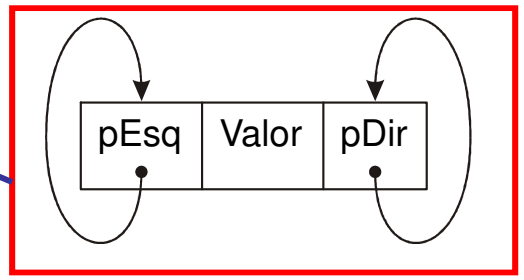


Lista Duplamente Encadeada

Árvore Binária

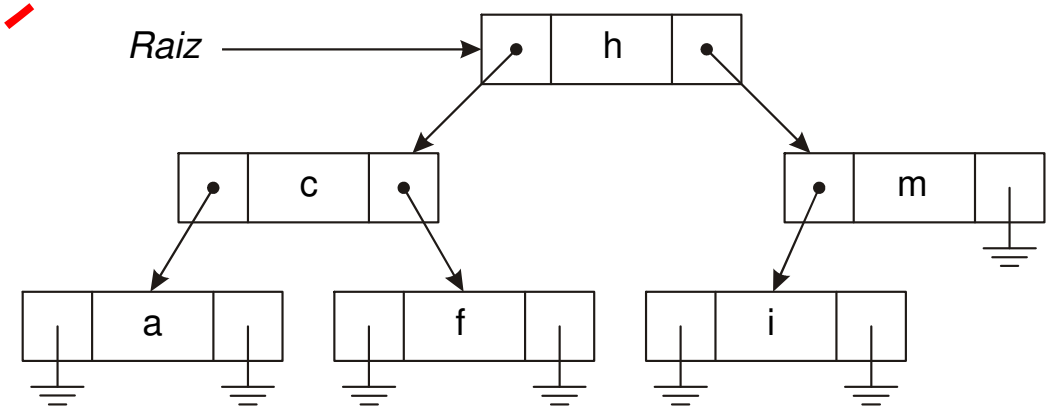
Quais problemas você identifica neste modelo?

Modelo de árvore binária



exemplo válido

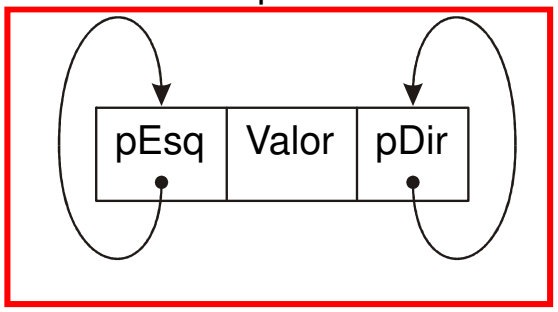
Exemplo de árvore binária



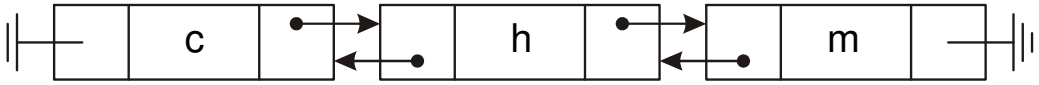
Cuidado com Figuras

Use descrição de assertivas complementares

Modelo de lista duplamente encadeada



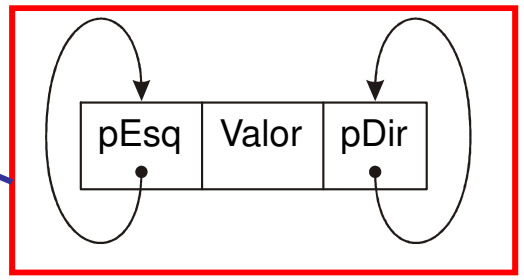
Exemplo de lista duplamente encadeada



Árvore Binária

Quais problemas você identifica neste modelo?

Modelo de árvore binária



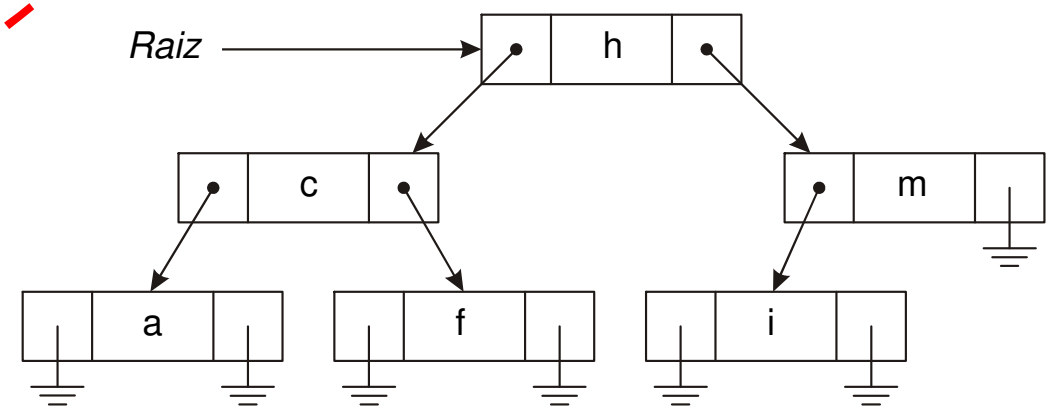
exemplo válido

Lista Duplamente Encadeada

1 pEsq == NULL

1 pDir == NULL

Exemplo de árvore binária

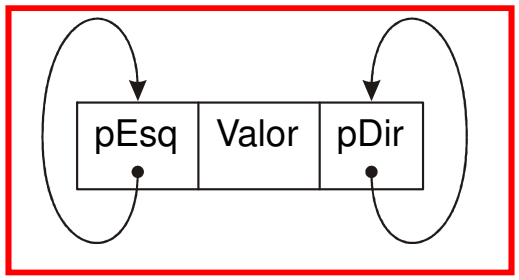


Figuras não são suficientes

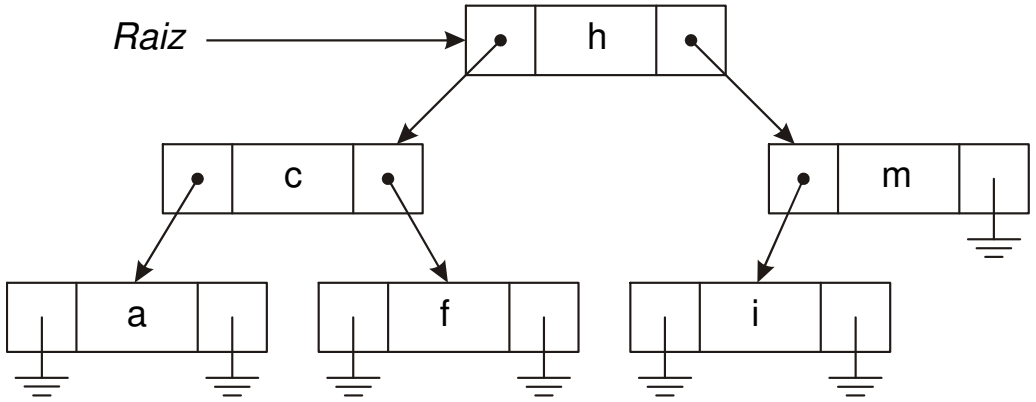
Árvore Binária

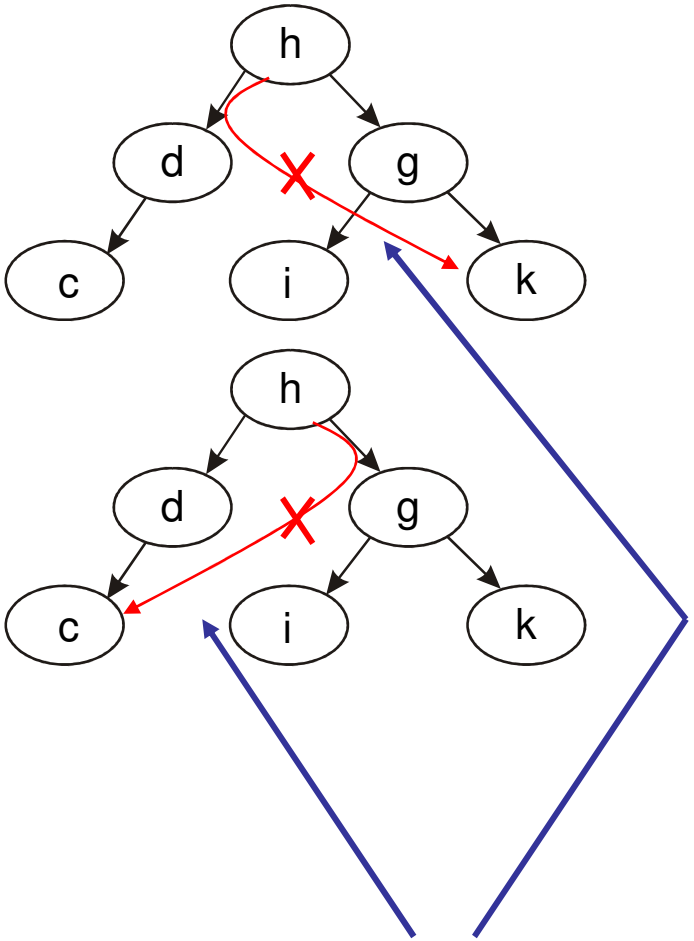
Quais problemas você identifica neste modelo?

Modelo de árvore binária



Exemplo de árvore binária





falta de detalhes importantes

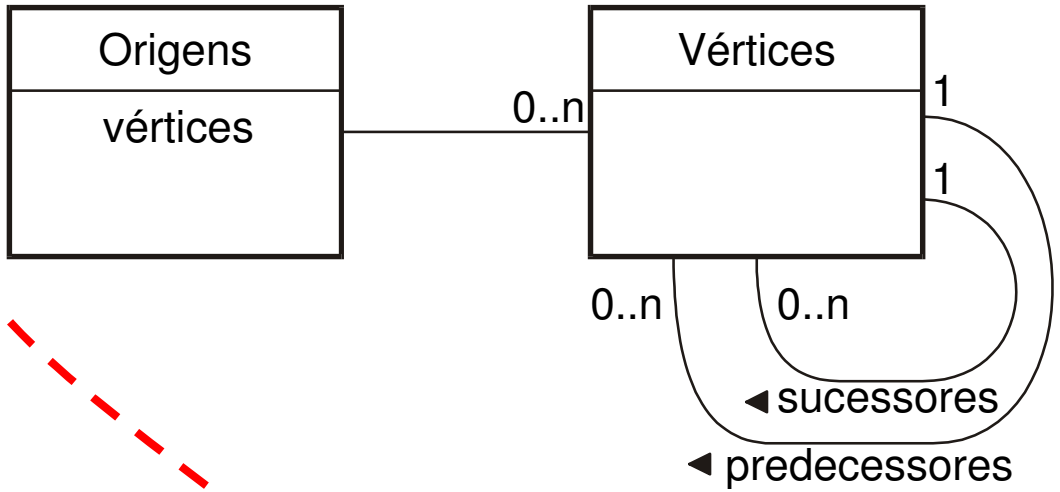
Modelos devem ter assertivas adicionais

- Para resolver o problema da insuficiência de detalhes em figuras utilizam-se assertivas estruturais.
- Exemplo: lista duplamente encadeada
  - Para cada nó N da lista
    - se N->pEsq != NULL então N->pEsq->pDir == N
    - se N->pDir != NULL então N->pDir->pEsq == N
- Exemplo: árvore
  - para cada nó N da árvore
    - a referência para filho à esquerda de um nó tem como destino a raiz da sub-árvore à esquerda
    - a referência para filho à direita de um nó tem como destino a raiz da sub-árvore à direita
    - o conjunto de nós alcançáveis a partir da raiz da sub-árvore à esquerda é disjunto do conjunto de nós alcançáveis a partir da raiz da sub-árvore à direita

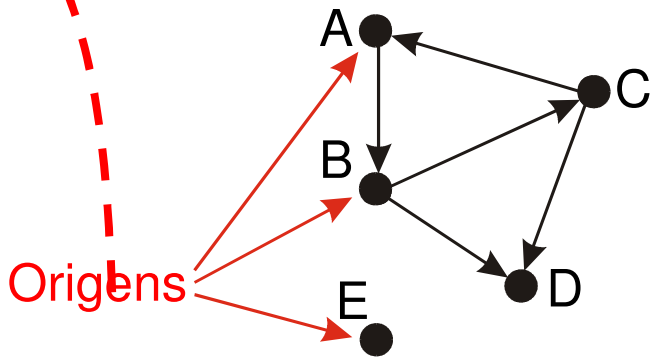
Outro exemplo: Grafo Dirigido

Modelo Físico
(um possível)

?

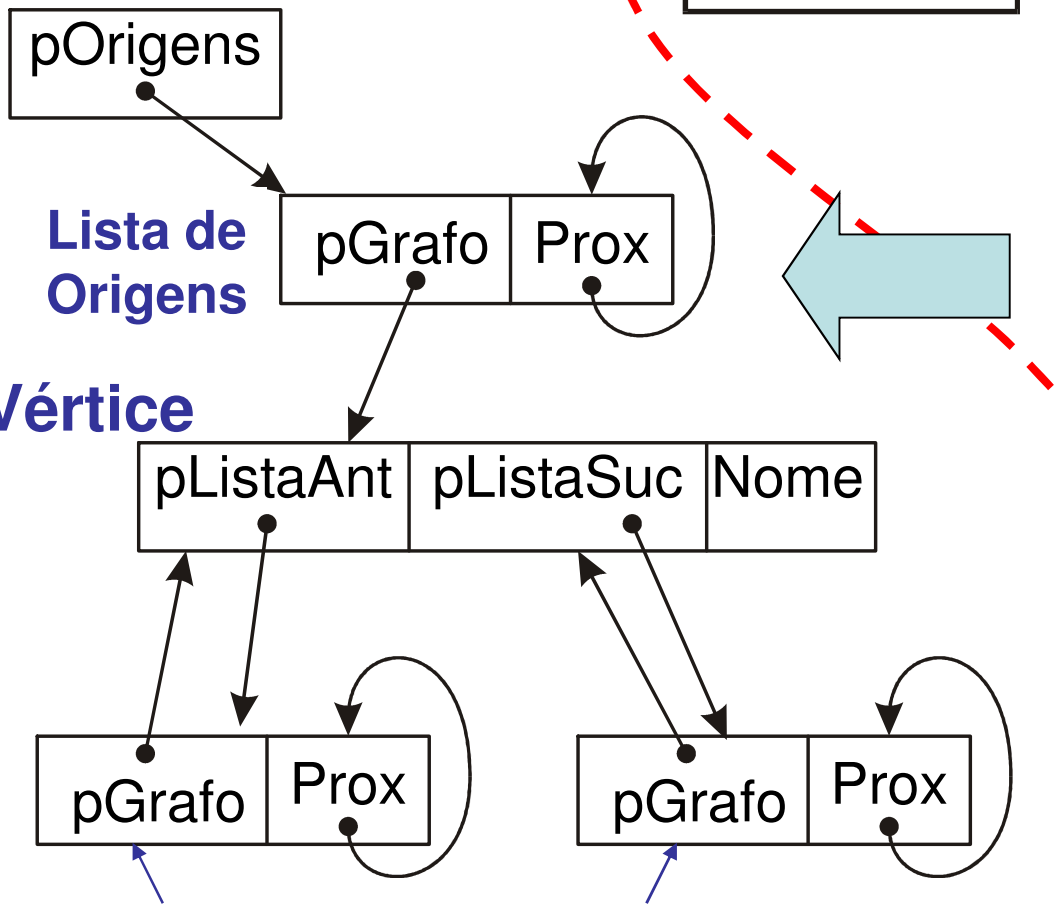


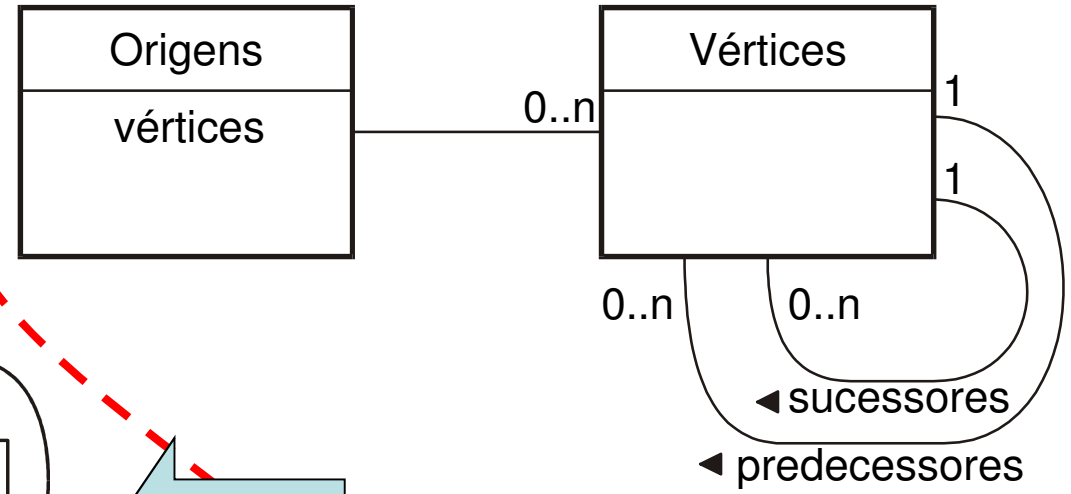
Modelo Conceitual



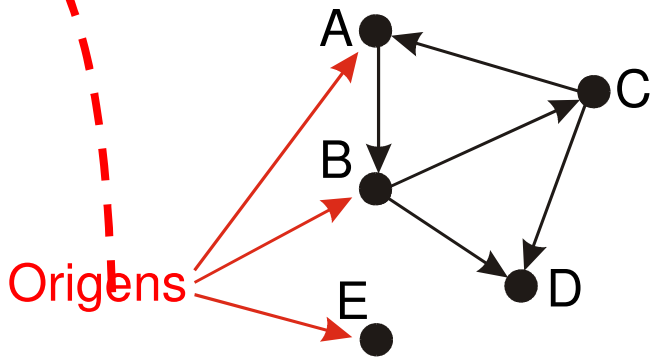
Outro exemplo: Grafo Dirigido

Modelo Físico
(um possível)

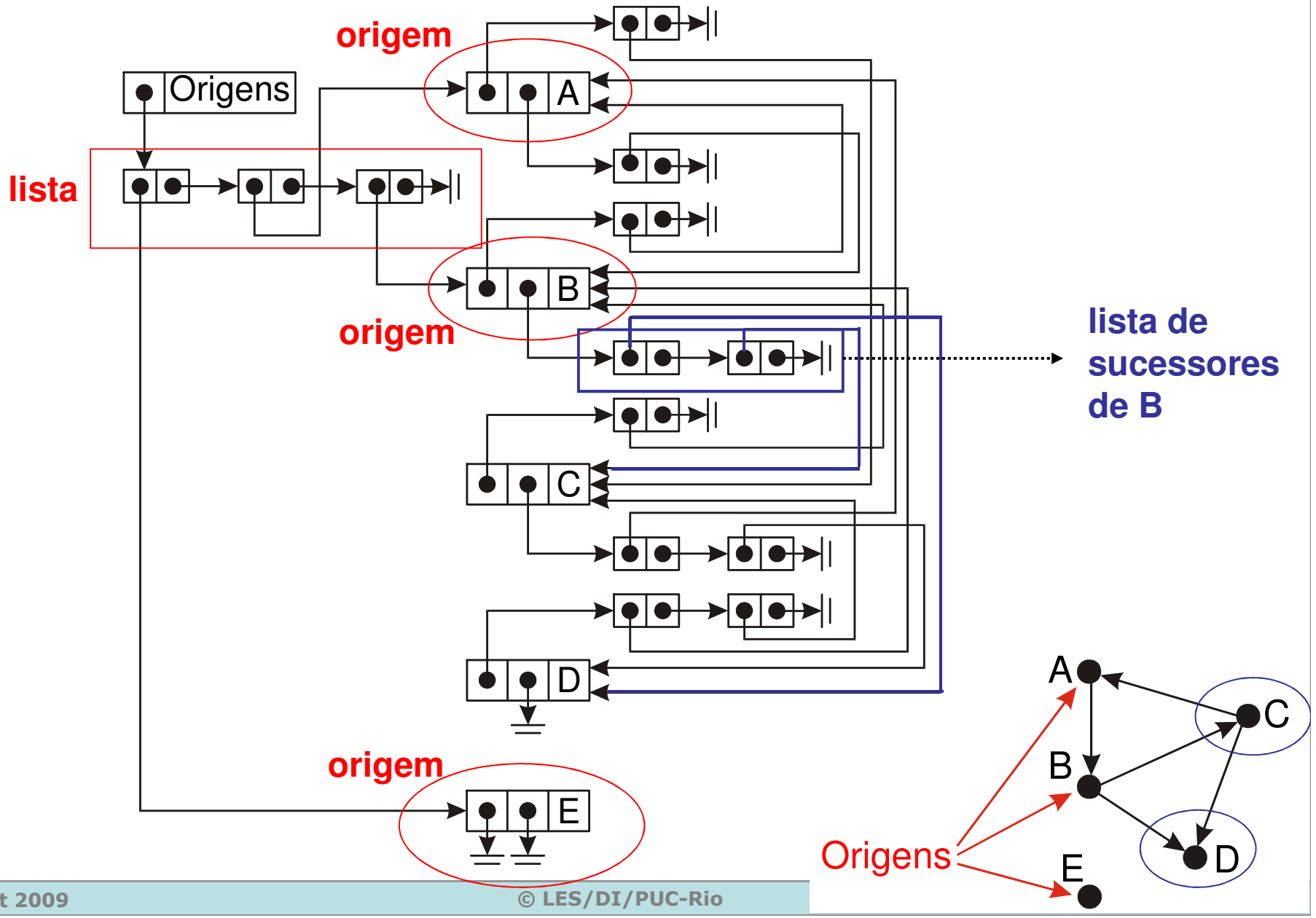




Modelo Conceitual



Exemplo físico do grafo



Correção do Questionário

- Objetivo:
  - identificar "padrões de faltas" na interpretação de perguntas
  - discutir *problemas recorrentes com certos conceitos* de programação modular

- Uso de exemplos concretos (de forma anônima) de respostas incompletas

- Não esqueçam de verificar se minha somatória foi correta

Interpretação incompleta da pergunta

- 1) O que é um módulo? Quais as duas características ...?

omissões comuns

- 3) Por que desenvolvemos programas de forma modularizada? Explique e compare *brevemente* acoplamento, coesão e encapsulamento.

- 7) Explique o que é análise estática e análise dinâmica. Dê **exemploS** de como revisões e testes se complementam. (3 pts)

Questão 1

- 1) O que é módulo?
  - "É uma parcela do código do programa que pode compilado separadamente."
  - "É a implementação de um conceito ou objeto da vida real."

(propriedade física)

E

(propriedade lógica)

Por que são incompletas?

Questão 2

- 2) Por que desenvolvemos programas de forma modularizada? Explique e compare *brevemente* acoplamento, coesão e encapsulamento. (2 pts)
  - Várias boas respostas: Waldecir, Daniel Castro, Aloan, etc.....

- 3) Explique falta/erro/falha

- Várias boas respostas: Isabelle, Daniel Castro, Waldecir, Gabriel Braga, etc.....

Questões 4 e 5

- 4) O que é programa correto? É possível criar um programa correto sem especificação? (1 pt)

  Várias respostas corretas...

  - Sheriton, Luis Paulo Mattos, Rodrigo, Raphael, Nicollas, e vários outros

- 5) Quais as diferenças entre teste manual e teste automatizado? (1 pt)

  Várias respostas corretas...

  - Raphael, Waldecir, Rodrigo, Sheriton, Breno, Gabriel Coelho, Paulo Henrique Alves, Thiago Motta, e vários outros

Outras questões

- Cuidados especiais
  - Análise estática e dinâmica não servem somente para detecção de faltas!
  - Requisito não-funcional não tem haver somente com 'desempenho'
    - É qualquer atribute de qualidade que influenciam várias decisões de projeto, processo e implementação
  - Outros problemas são mencionados na minha correção

Dúvidas - Trabalho

- Alguns de vocês tem enviado exemplos de structs, perguntando se está correto?
  - sem antes ler com atenção o enunciado e as instruções da aula de requisitos e modelagem da arquitetura

```
typedef struct LIS_Cabeca
{
    LIS_tpUsuario * pIniLista ;
    LIS_tpUsuario * pFinal ;
    LIS_tpUsuario * pCorrente ;

} LIS_tpCabeca ;

typedef struct LIS_Usuarios
{
    void * Valor ;
    LIS_tpUsuario * pProx ;
    LIS_tpUsuario * pAnt ;
    LIS_tpRef * pArestas ;

} LIS_tpUsuario ;
```

Dúvidas - Trabalho

- Alguns de vocês tem enviado exemplos de structs, perguntando se está correto?
  - sem antes ler com atenção o enunciado e as instruções da aula de requisitos e modelagem da arquitetura
- Passos
  - **1)** Elicite os requisitos
    - Quais as funcionalidades do programa da rede de relacionamentos?
    - Estudar o que vem a ser um grafo? Caso você não saiba o que vem a ser um grafo. Isso faz parte da elicitação de requisitos.
  - Os requisitos serão importantes para que vocês descubram:
    - Módulos, interfaces e dependências
    - Quais outras características (estrutura/ funções) dos módulos Lista e Grafo?

```
typedef struct LIS_Cabeca
{
    LIS_tpUsuario * pIniLista ;
    LIS_tpUsuario * pFinal ;
    LIS_tpUsuario * pCorrente ;

} LIS_tpCabeca ;

typedef struct LIS_Usuarios
{
    void * Valor ;
    LIS_tpUsuario * pProx ;
    LIS_tpUsuario * pAnt ;
    LIS_tpRef * pArestas ;

} LIS_tpUsuario ;
```

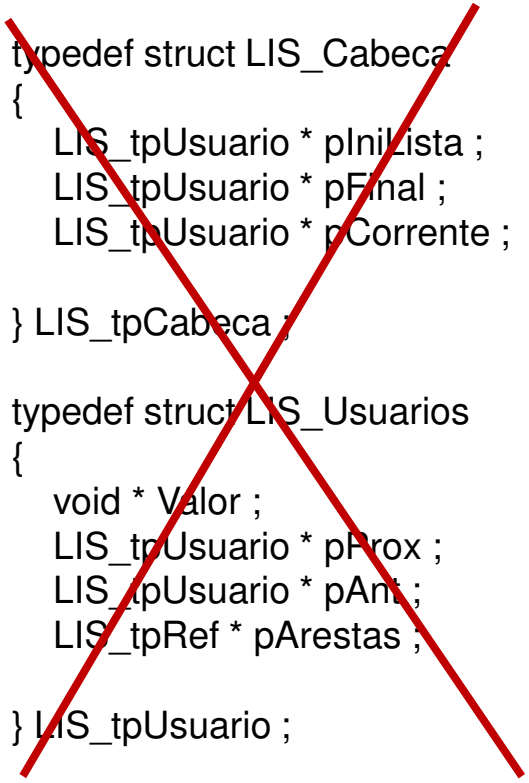
Dúvidas - Trabalho

- Alguns de vocês tem enviado exemplos de structs, perguntando se está correto?
  - sem antes ler com atenção o enunciado e as instruções da aula de requisitos e modelagem da arquitetura
- Passos
  - **2)** Modele a arquitetura
    - .. usando notação de modelo de componentes
  - **3)** Aí sim, defina modelo físico com as estruturas de dados necessárias
    - modelagem do Grafo e outras estruturas necessárias da rede de relacionamentos
    - enunciado já diz quais são as estruturas que vão compor o Grafo!
    - adicione as estruturas da rede de relacionamentos (depende dos requisitos!)

```
typedef struct LIS_Cabeca
{
    LIS_tpUsuario * pIniLista ;
    LIS_tpUsuario * pFinal ;
    LIS_tpUsuario * pCorrente ;

} LIS_tpCabeca ;

typedef struct LIS_Usuarios
{
    void * Valor ;
    LIS_tpUsuario * pProx ;
    LIS_tpUsuario * pAnt ;
    LIS_tpRef * pArestas ;

} LIS_tpUsuario ;
```

Dúvidas - Trabalho

- Alguns de vocês tem enviado exemplos de structs, perguntando se está correto?
  - sem antes ler com atenção o enunciado e as instruções da aula de requisitos e modelagem da arquitetura
- Passos
  - **4)** Modifique (se necessário) e teste o modelo Lista genérica
    - para testar, crie módulo a parte ou uso o módulo de Teste
  - **5)** Implemente e teste o módulo Grafo genérico conforme enunciado



Dúvidas - Trabalho

- Para que serve a função IrVertice?
  - pode implementá-la como uma função que procura um valor, assim como a função ProcuraValor do módulo Lista
  - depende, de certa forma, da especificação de requisitos
    - note que nem todas funções do módulo Grafo genérico precisarão necessariamente ser usado por outros módulos
  - pode ser que seus requisitos definam diferentes funcionalidades de busca na rede de relacionamentos
    - um nome de usuário
    - um endereço
    - um telefone
    - Etc…
    - Logo: pode ter que usar ponteiros de funções que apontem para diferentes funções de comparação
      - a ser visto na aula sobre Polimorfismo

Dúvidas - Trabalho

- É necessário **lista de origens**?
  - Vide enunciado: não é obrigatório; novamente: depende dos requisitos
  - Pode ser interessante: se usuários entram sem serem convidados (criando grafos desconexos)
- Considerar que é um **grafo dirigido**?
  - Grupo decide!
  - Também depende dos seus requisitos
    - Por exemplo, se deve manter quem convidou quem…
    - Direção modela/informa quem fez o convite
- Cada **vértice deve guardar sua lista de arestas**?
  - Já é dito no enunciado
- E se for necessário **manter informações sobre arestas** (por exemplo, data/horario de criação do relacionamento)?
  - Pode criar uma struct que será armazenada no valor da lista de arestas
    - Ex. Ponteiro para o Vertice (aresta) e infos do relacionamento
  - **Importante**: esta é estrutura é específica da rede de relacionamentos e não deve ser incluída no módulo Grafo!

Todas assertivas estruturais e funções...

- … mencionadas no enunciado do Trabalho
  - são um conjunto mínimo para a definição das structs e das funções
  - você pode decidir adicionar outros elementos as structs, se necessário, conforme você especificar as funcionalidades dos requisitos:
    - ex. elemento Marcado que indica que um Vértice ou Aresta já foi visitado (usado, por exemplo, por uma função que mostra todos os usuários e relacionamentos)
  - não necessariamente vai precisar usar nos módulos clientes todas as funções de Lista e Grafo

Itens que foram "ignorados" na correção...

- Do T1, mas que serão importantes no T2 em diante:
  - padrões de programação
    - documentação das interfaces e da implementação
    - política de escolha de nomes
    - organização do código
    - definição de constantes
    - Etc…
  - assertivas (principalmente no módulo Grafo)
    - executáveis: pré-condições e pós-condições das funções
    - comentários: assertivas estruturais (por exemplo, que ListaVertice só tem 1 elemento)
      - alternativamente, como descrição adicional aos modelos
      - idealmente, tanto no modelo como no código
  - uso apropriado de conversão/imposição de tipos
  - e outras matérias dadas até aqui…

Dúvidas - Trabalho

- ***Como elicitar* os requisitos não-funcionais e saber se eles são *completos*?**
  - Aula passada: vários exemplos foram dados
    - identifique aqueles que são relevantes para o desenvolvimento do programa de rede de relacionamentos
  - O livro provê um catálogo de requisitos não-funcionais (cáp. 10.1)
    - mas, no mínimo, focalize naqueles que tem sido amplamente discutidos durante as aulas do curso
      - quais são?
  - Exemplos
    - **Privacidade**:
      - um usuário não pode ter informações sobre outro usuário que só permite que seus relacionados tenham tais informações
    - **Reusabilidade, Manutenibilidade, Robustez**, etc...

Dúvidas - Trabalho

- **Para os requisitos funcionais: basta as regras da rede de relacionamentos?**
  - Não! devem definir as funcionalidades requeridas que permitem consultas e outras funcionalidades da rede de relacionamentos
  - ... a serem implementadas no T3
    - Como descobri-las? Vide aula passada: perguntas e respostas
    - Análise de domínio: softwares de rede de relacionamentos... orkut, facebook, etc...

Dúvidas - Trabalho

- **Existem outros módulos na arquitetura além de Grafo e Lista?**
  - Sim! Lembre que deve ser a arquitetura do programa que permite gerenciar a rede de relacionamentos
    - Como descobrir tais módulos?
      - Reflita sobre as descrições de todos requisitos...
        - "o sistema deve permitir criar novos usuários..." (gerencia da rede)
        - "uma interface permite indicar quais buscas podem ser feitas na rede de relacionamentos..."

Dúvidas - Trabalho

- **Modelos Físicos**
  - Não esqueça de ***descrever assertivas*** que não podem ser expressas visualmente nos modelos
  - Não esqueça de entregar (último item) pelo menos um **exemplo físico**
  - **Não é obrigatório representar o ponteiro para função** de remoção de elementos da lista

Lembrete – Prova, Revisão e Monitoria

- **Prova**: 29 de setembro, próxima quarta, horário de aula
  - 13:00 às 14:50
  - **começa as 13:00 em ponto**
    - **logo, recomenda-se, se possível, chegar 5 minutos antes**
  - com consulta, não é permitido uso de laptops e fazer perguntas sobre interpretação da pergunta
  - **local**: sala de aula
  - **matéria:** até hoje
- **Dúvidas da matéria:**
  - Envie email para mim ou Francisco
  - Lembrete, próxima segunda (27/09) – aula de Revisão junto com a turma do Prof. Flávio, horário: 17-19, sala L504
- Monitoria com Prof. Francisco:
  - De 25/09 a 25/10, será por skype. Marque horário com antecedência.

Perguntas/dúvidas Específicas

- Questões de prova poderão ser respondidas em C++/Java?
  - De preferência para respostas em C
    - afinal, foi a linguagem utilizada nos trabalhos
  - O objetivo deste curso não é um curso específico de uma linguagem de programação; portanto:
    - em certas aulas, usamos exemplos em diferentes linguagens (p.e. C++/Java) para ilustrar que conceitos/princípios se aplicar a programas em qualquer linguagem
    - raramente se pedirá para você escrever código na prova
      - logo, esta questão se torna praticamente irrelevante para provas
  - Casos especiais:
    - obviamente, certos padrões de organizações de módulos ou padrões de faltas discutidos no curso são específicos para C
      - por exemplo, separação entre interface e implementação: *.h vs. *.c

Questões sobre Modelos

- **Evite "inventar"** elementos da **notação** de modelagem
  - Por exemplo, cada triângulo é uma interface...
    - assim que lê sua especificação não sabe qual foi a intenção do modelador
  - Na pior das hipóteses, crie uma legenda para comunicar a semântica de cada elemento da "sua notação"
- Caso você for usar diferentes estilos de setas para expressar diferentes de relacionamentos entre módulos, **use legendas**...
- **Cuidado com uso errado da notação**, por exemplo:
  - referência ao invés de usar agregação
- **Não esqueça de dar nomes** aos elementos do modelo: nomes dos módulos, relacionamentos, das interfaces, etc...

Dicas gerais

- Se houverem questões do tipo:
  - "Dado o código/modelo abaixo, redija um texto sobre os princípios de de modularidade seguidos ou violados."
- Busque **completude** e coerência na sua resposta, mas escreva **somente o necessário**
  - **Evite divagar** sobre o assunto
  - Não fantasie e escreva sobre o que você não tem certeza do que está falando
  - Tamanho da resposta não é proporcional à nota
- Seja preciso e maximize o uso de **terminologia básica da Programação Modular**
  - por exemplo, use: acoplamento, interface, encapsulamento, assertivas, padrão de programação, faltas, erros, falhas, etc...
  - evite termos demasiadamente genéricos ou não definidos/relevantes neste curso, exemplo: coisa, classe,

Dicas gerais

- Justifique suas respostas!
  - Evite respostas vagas e sem evidências do tipo
    - "Sim, a função X viola padrões de programação."
    - Prefira respostas com argumentação e precisão apropriada:
      - "A 5ª linha de código da função X viola o padrão de identificadores X porque o prefixo do nome da variável Z...."
      - "A expressão A na linha 3 exibe um padrão de falta clássico pois...."
      - "O uso de malloc e sizeof na linha 7..."

Aula 15

Modelagem Física

Alessandro Garcia
LES/DI/PUC-Rio
Setembro 2010