Universidade Federal de Mato Grosso do Sul
Sistemas de Informação - CPCX

Diagrama de Classes – Aula 11 (parte 1)

Prof. Fernando Maia da Mota

Slides gentilmente cedidos por Profa. Dra. Maria Istela Cagnin Machado
UFMS/FACOM

Diagrama de Classes

- Segundo o PU, na última etapa da fase de projeto deve-se compor as classes de projeto do sistema que são as classes que serão efetivamente implementadas utilizando uma linguagem de programação orientada a objetos.

Diagrama de Classes

- Com base nos diagramas de comunicação e no modelo conceitual do sistema pode-se finalmente criar o diagrama de classes de projeto.

Visibilidade entre Objetos

- Visibilidade: capacidade de um objeto ver ou fazer referência a outro
  - Para que um objeto A envie uma mensagem para o objeto B, é necessário que B seja visível para A

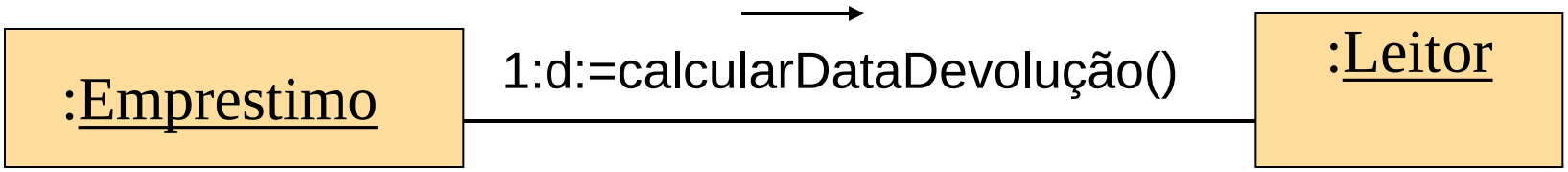
Tipos de visibilidade

- por atributo
  - Quando as classes de dois objetos estão associadas no modelo conceitual
- por parâmetro
  - Quando um objeto recebe outro como parâmetro de método
- localmente declarada
  - Quando um objeto recebe outro como retorno de método
- global
  - Quando um objeto é declarado em âmbito global

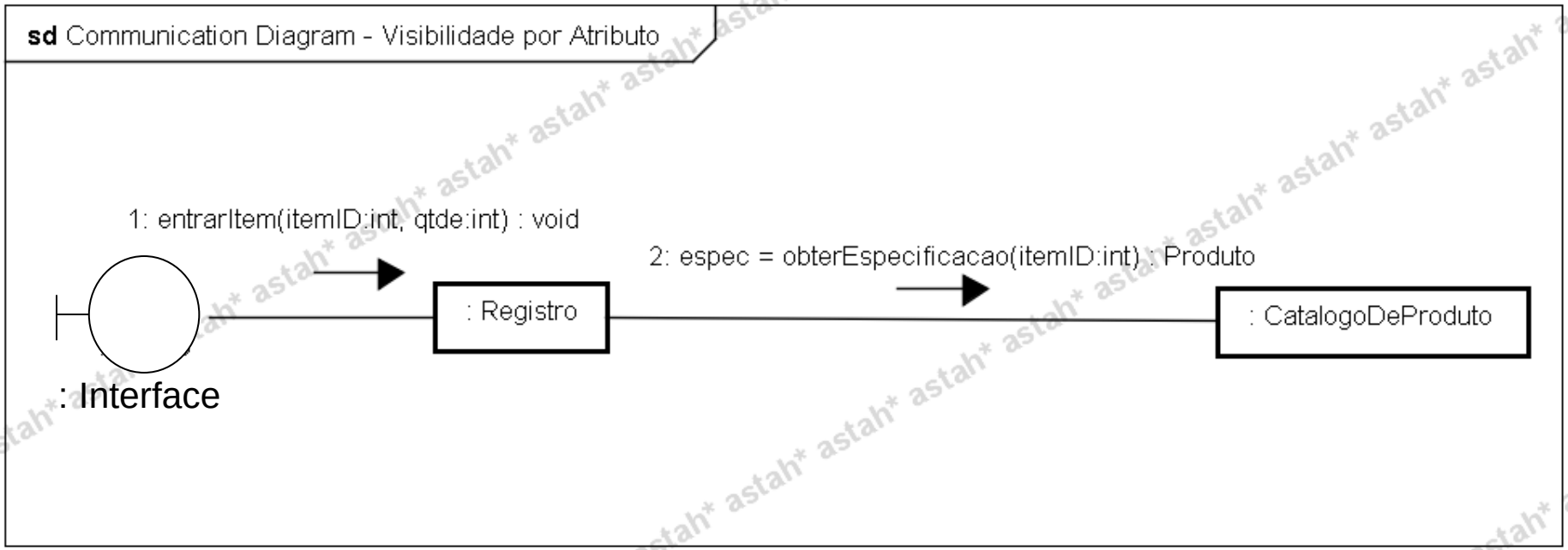
Visibilidade por atributo

- É a forma mais comum
- Geralmente se deve às associações existentes no modelo conceitual

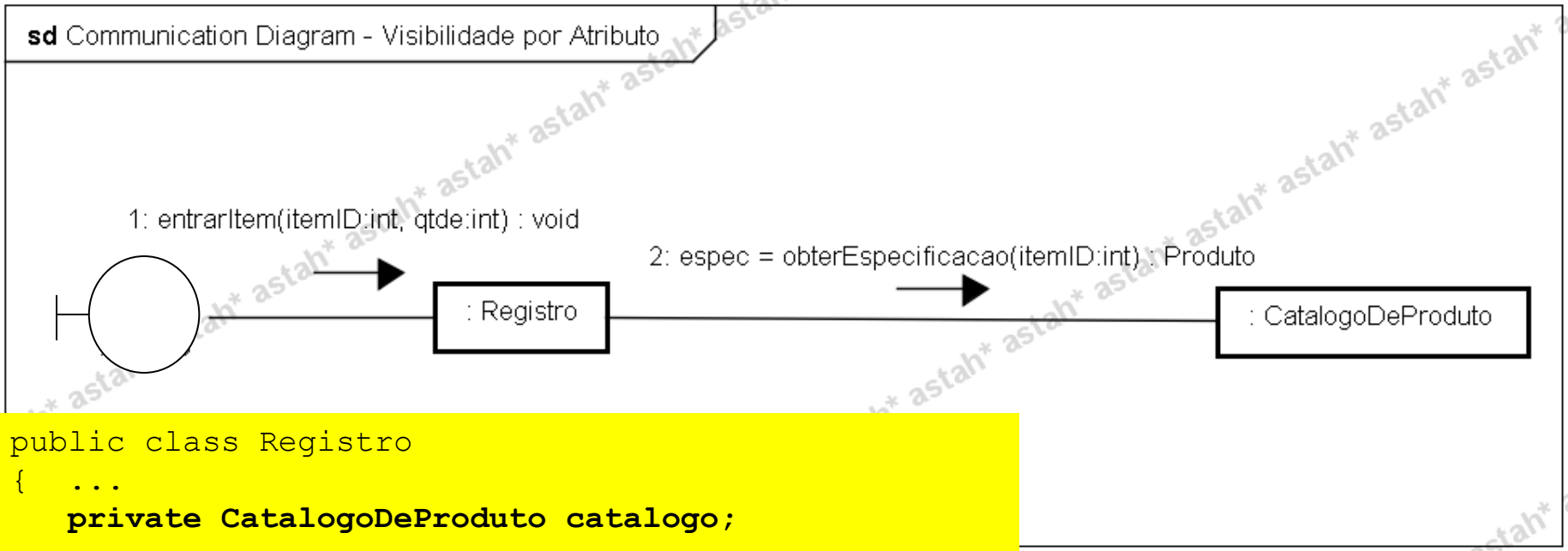
Exemplo: Visibilidade por Atributo



Exemplo: Visibilidade por Atributo



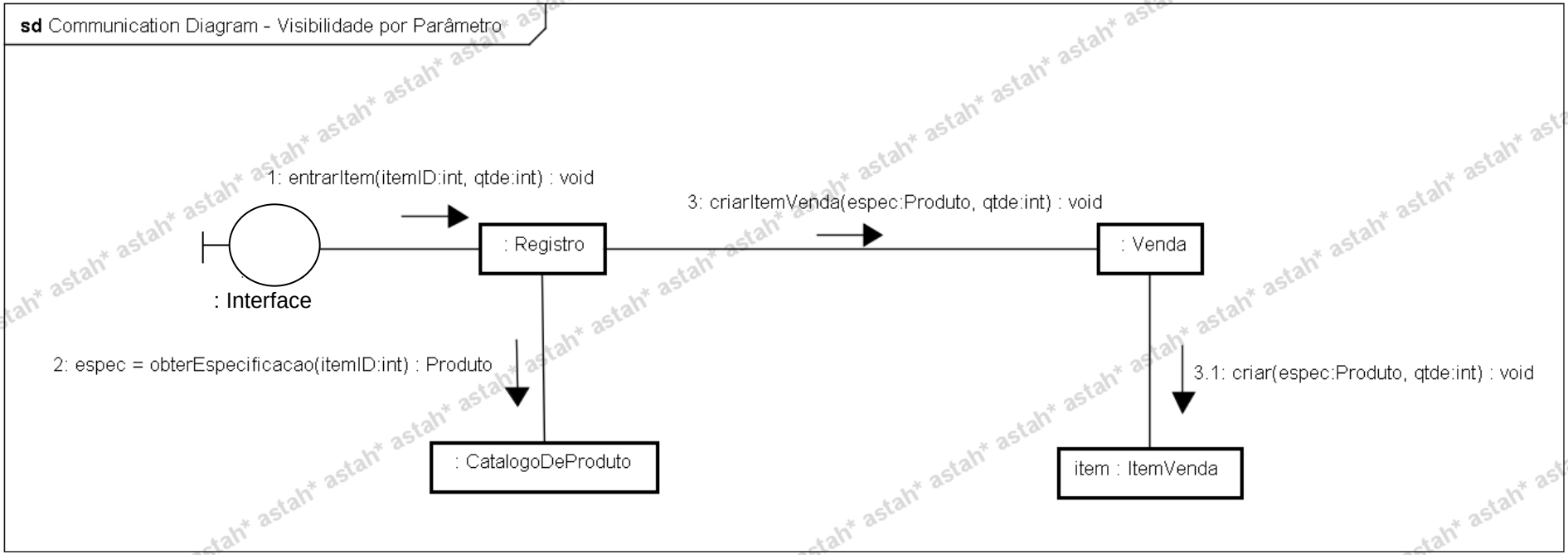
Exemplo: Visibilidade por Atributo



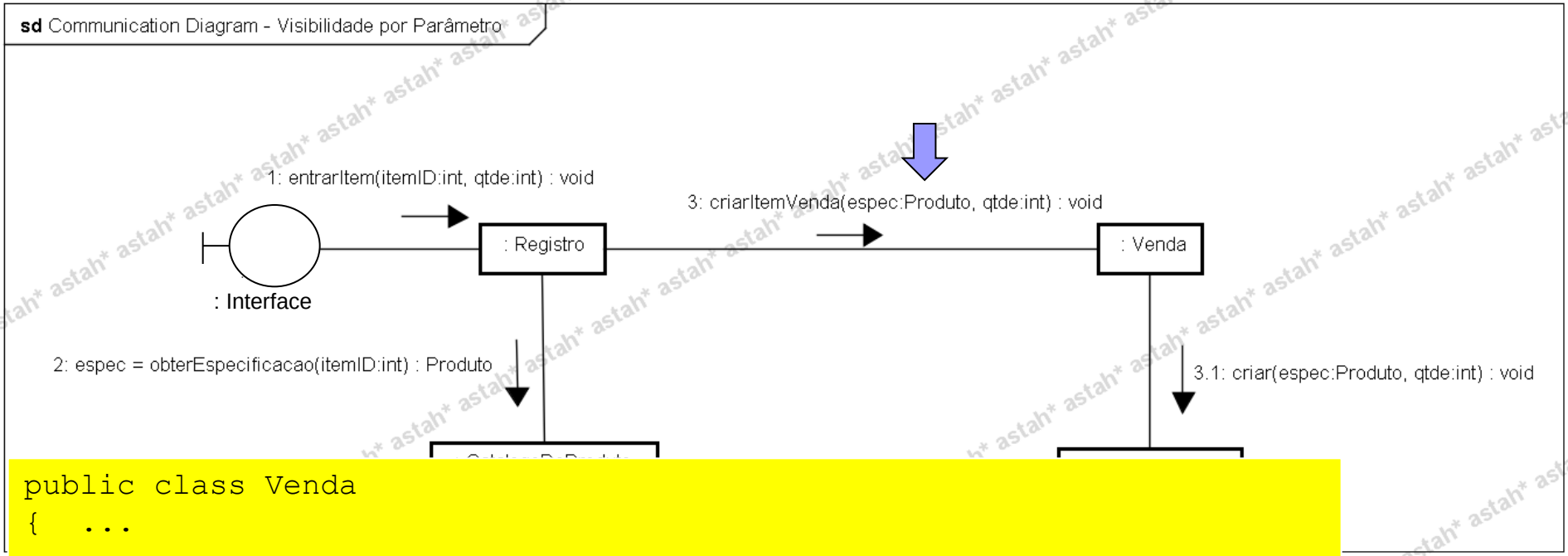

```
public class Registro
{   ...
    private CatalogoDeProduto catalogo;

    public void entrarItem(int itemID, int qtde)
    {
       ...
      espec = catalogo.obterEspecificacao(itemID);
        ...
     }
}
```

Exemplo: Visibilidade por Parâmetro



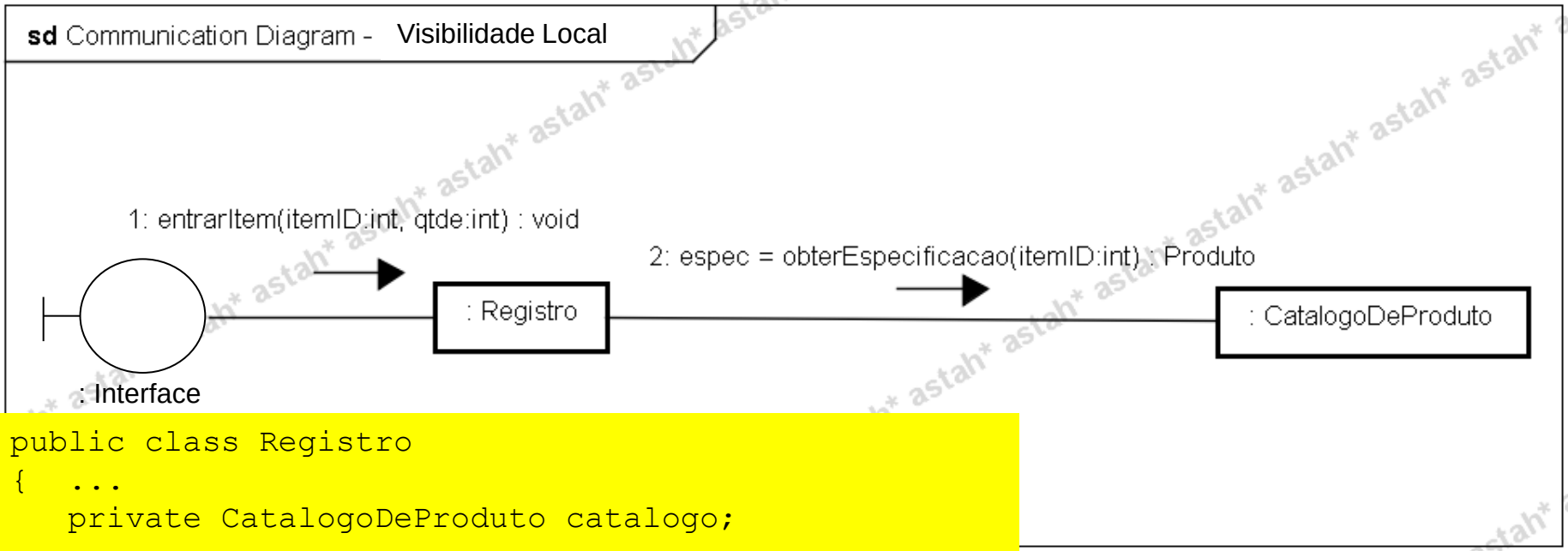
Exemplo: Visibilidade por Parâmetro




```
public class Venda
{  ...

   public void criarItemVenda(Produto espec, int qtde)
   {
     System.out.printl("Desc. Produto: " + espec.getDescricao());
     item = new ItemVenda(espec, qtde);
      ...
    }
}
```

Exemplo: Visibilidade Local




```
public class Registro
{   ...
    private CatalogoDeProduto catalogo;

    public void entrarItem(int itemID, int qtde)
    {
      Produto espec;
      espec = catalogo.obterEspecificacao(itemID);
       ...
     }
}
```

Diagrama de Classes de Projeto

- Informação típica:
  - classes, associações e atributos
  - métodos
  - tipos dos atributos
  - navegabilidade

Diagrama de Classes de Projeto

- Informação típica:
  - classes, associações e atributos
  - métodos
  - tipos dos atributos
  - navegabilidade

Modelo Conceitual x Diagrama de Classes de Projeto

- Modelo Conceitual $\Rightarrow$ abstrações de conceitos, ou objetos, do mundo real
  - conceitos são também chamados de classes conceituais

- Diagrama de Classes de Projeto $\Rightarrow$ definição de classes como componentes de software
  - classes de software

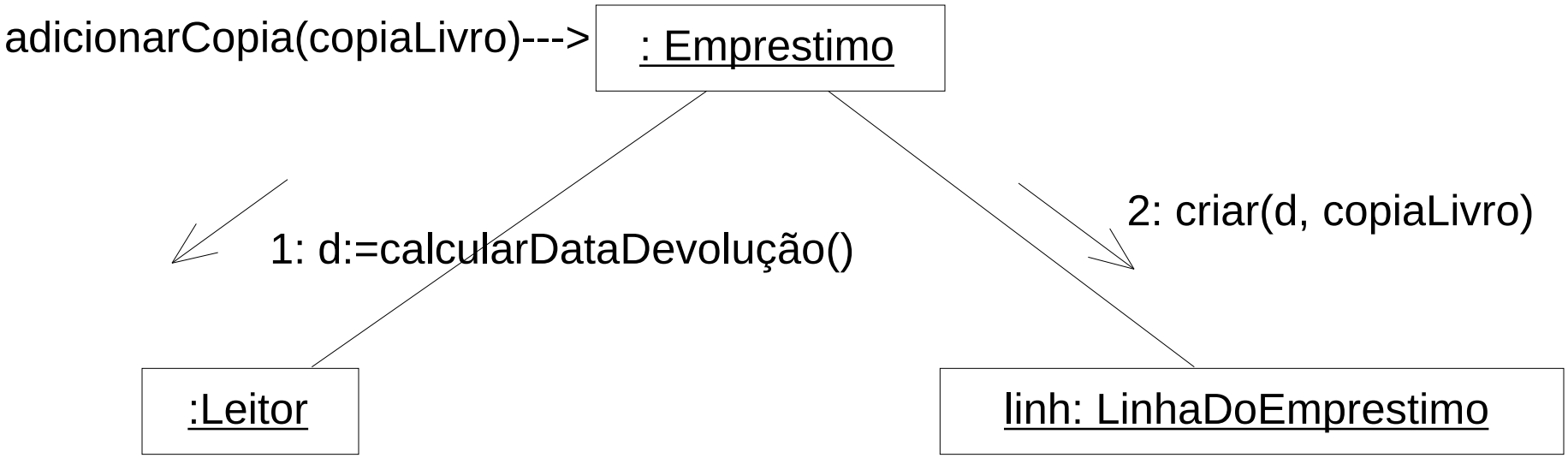
Diagrama de Classes de Projeto

- Em relação ao Modelo Conceitual, o DC apresenta:
  - Adição dos métodos
  - Adição da direção das associações (navegabilidade)
  - Possível detalhamento dos atributos e associações
  - Possível alteração na estrutura das classes e associações
  - Possível criação de atributos privados ou protegidos

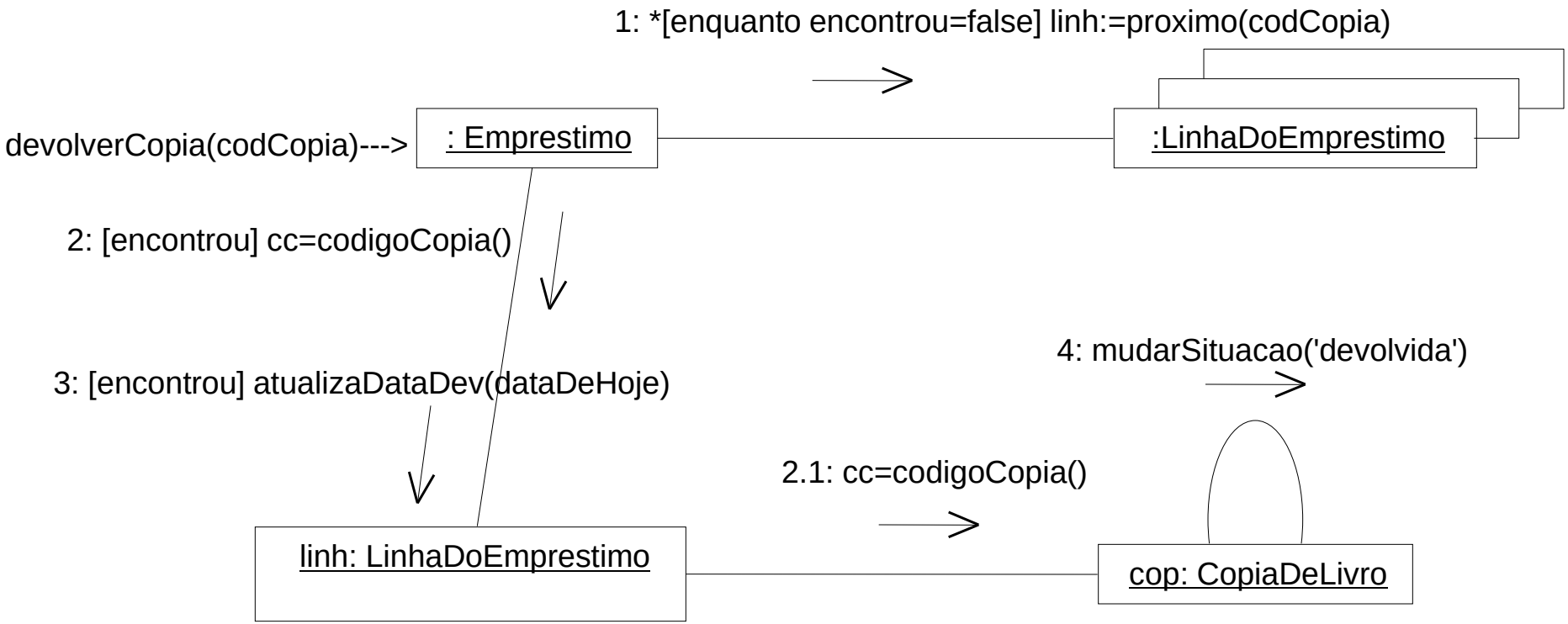
Definição

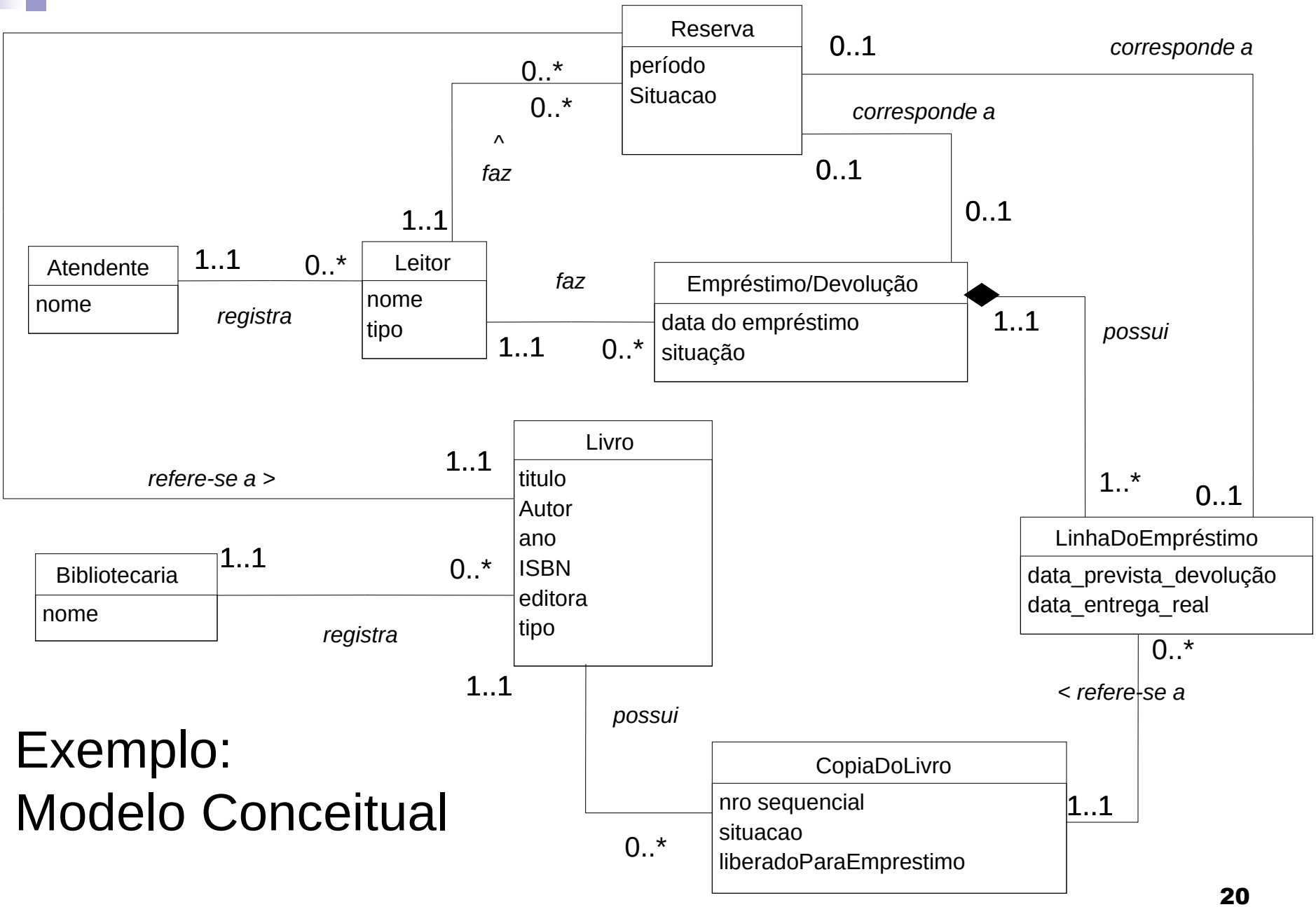
- Na prática, o Diagrama de Classes pode ser construído à medida que a fase de projeto avança, a partir dos diagramas de comunicação
  - Cada classe que aparece no diagrama de comunicação automaticamente é incluída no diagrama de classes de projeto
  - Os atributos são inicialmente os que estão no modelo conceitual

Exemplo: Diagrama de Comunicação 1



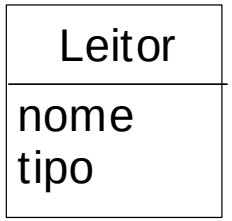
Exemplo: Diagrama de Comunicação 2

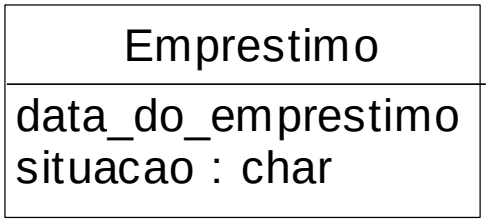


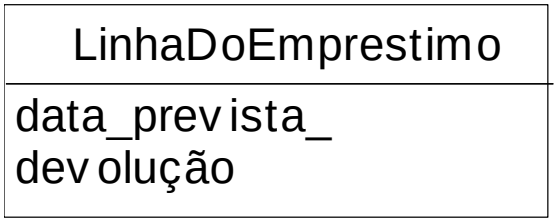


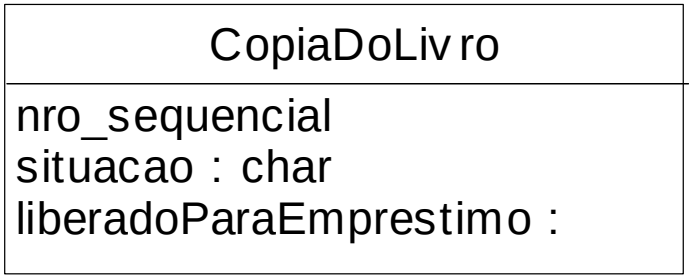
Exemplo: Modelo Conceitual

Classes que aparecem nos 2 diagramas de comunicação









Associações e Navegabilidade

- Associações e navegabilidade entre classes são indicadas pelos diagramas de comunicação
  - Navegabilidade indica possibilidade de navegação unidirecional por meio de uma associação entre classes
    - geralmente implica visibilidade por atributos
- Notação: seta contínua

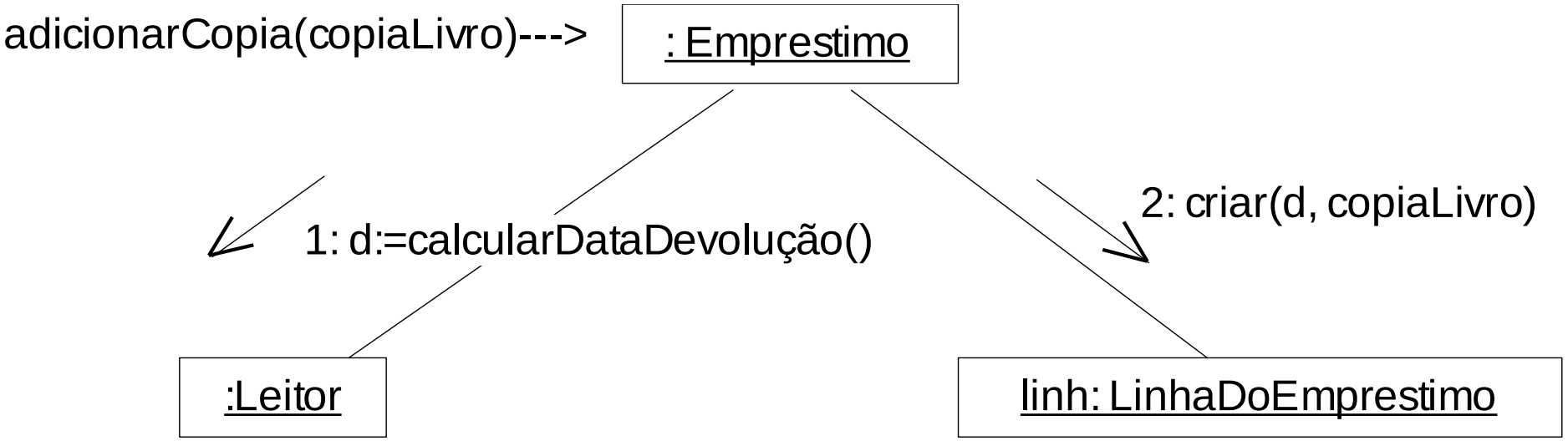
Associações e Navegabilidade

- Indícios de associação e com presença de navegabilidade:
  - A envia mensagem para B
  - A cria B
  - A precisa manter uma conexão com B

Como determinar a navegabilidade?

- Verificar o envio de mensagens de objetos que possuem visibilidade por atributo
- Desenhar a seta no sentido da classe que envia a mensagem para a classe que recebe a mensagem

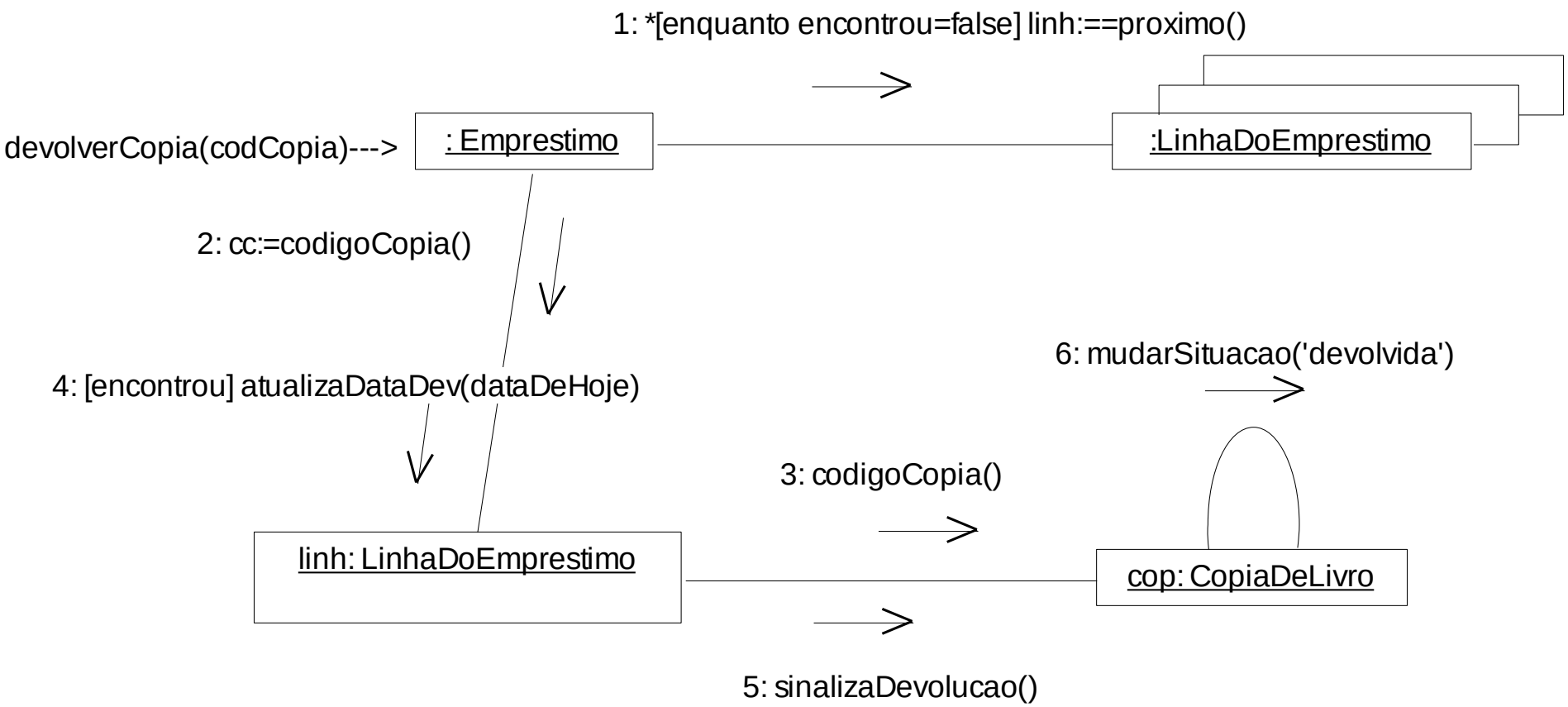
Navegabilidade



Este diagrama de comunicação implica nas navegabilidades:

Emprestimo -> Leitor

Emprestimo -> LinhaDoEmprestimo

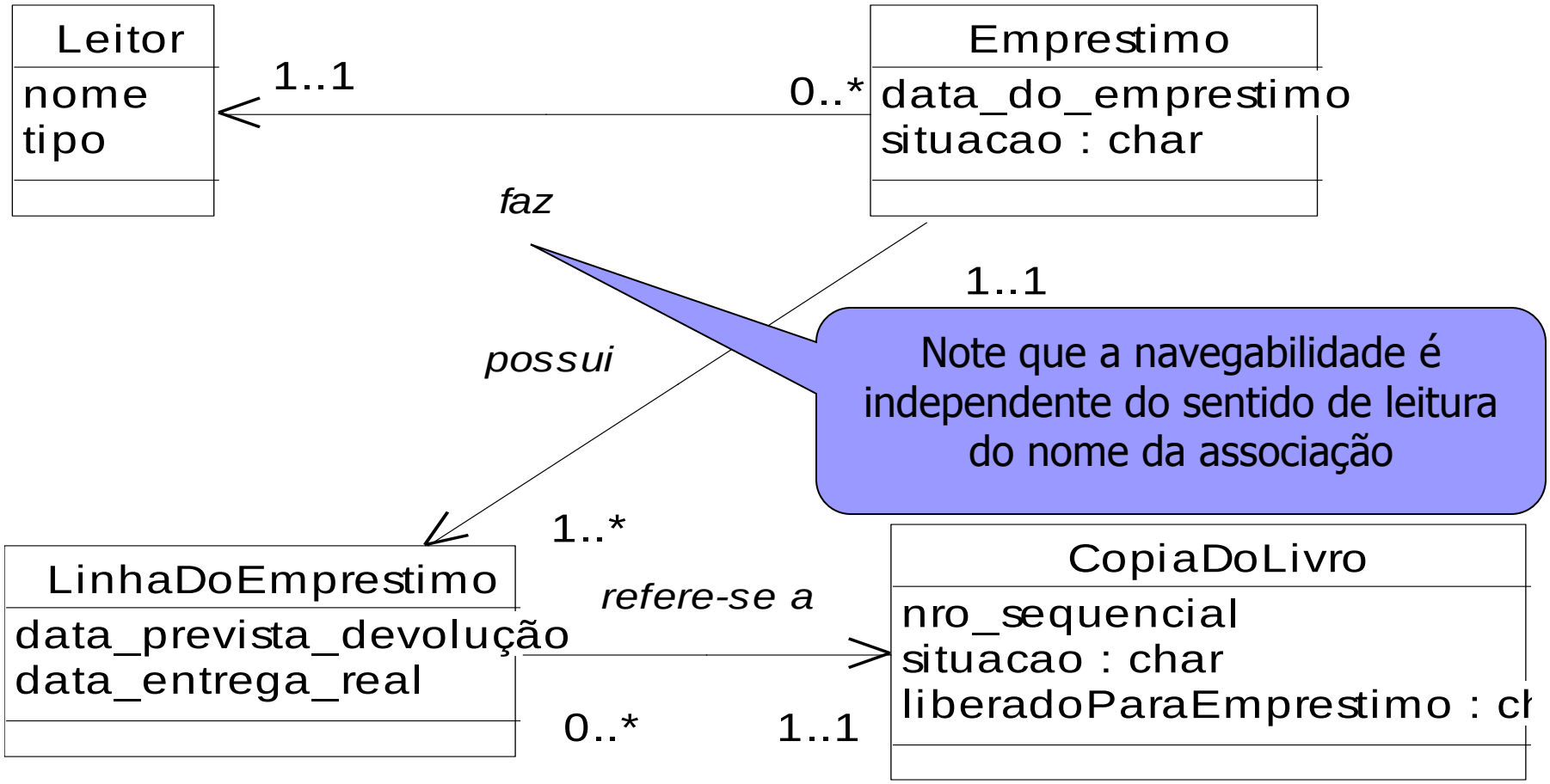


Este diagrama de comunicação implica nas navegabilidades:

Emprestimo -> LinhaDoEmprestimo

LinhaEmprestimo -> CopiaDoLivro

Diagrama de Classes com navegabilidade



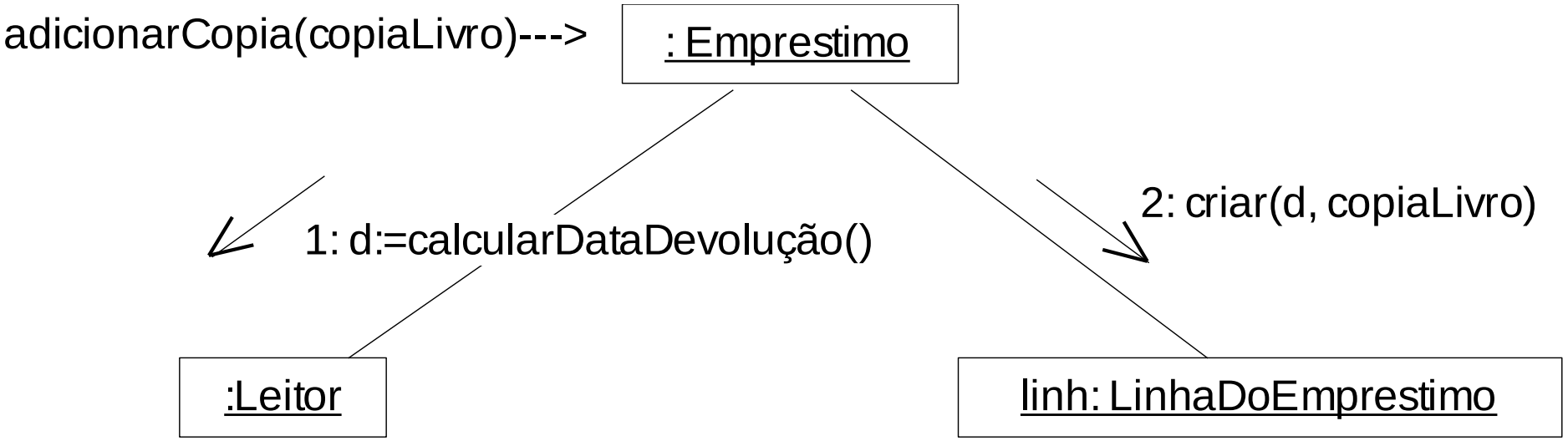
(com base apenas nos 2 diagramas de comunicação mostrados)

Como incluir os métodos nas classes?

- Métodos são incluídos nas classes que recebem a mensagem

- Linguagens de programação distintas podem ter sintaxes distintas para métodos
  - recomendável: usar sintaxe básica UML
    nomeMétodo(Par_1*,* Par_2*, … *Par_n*): retorno*

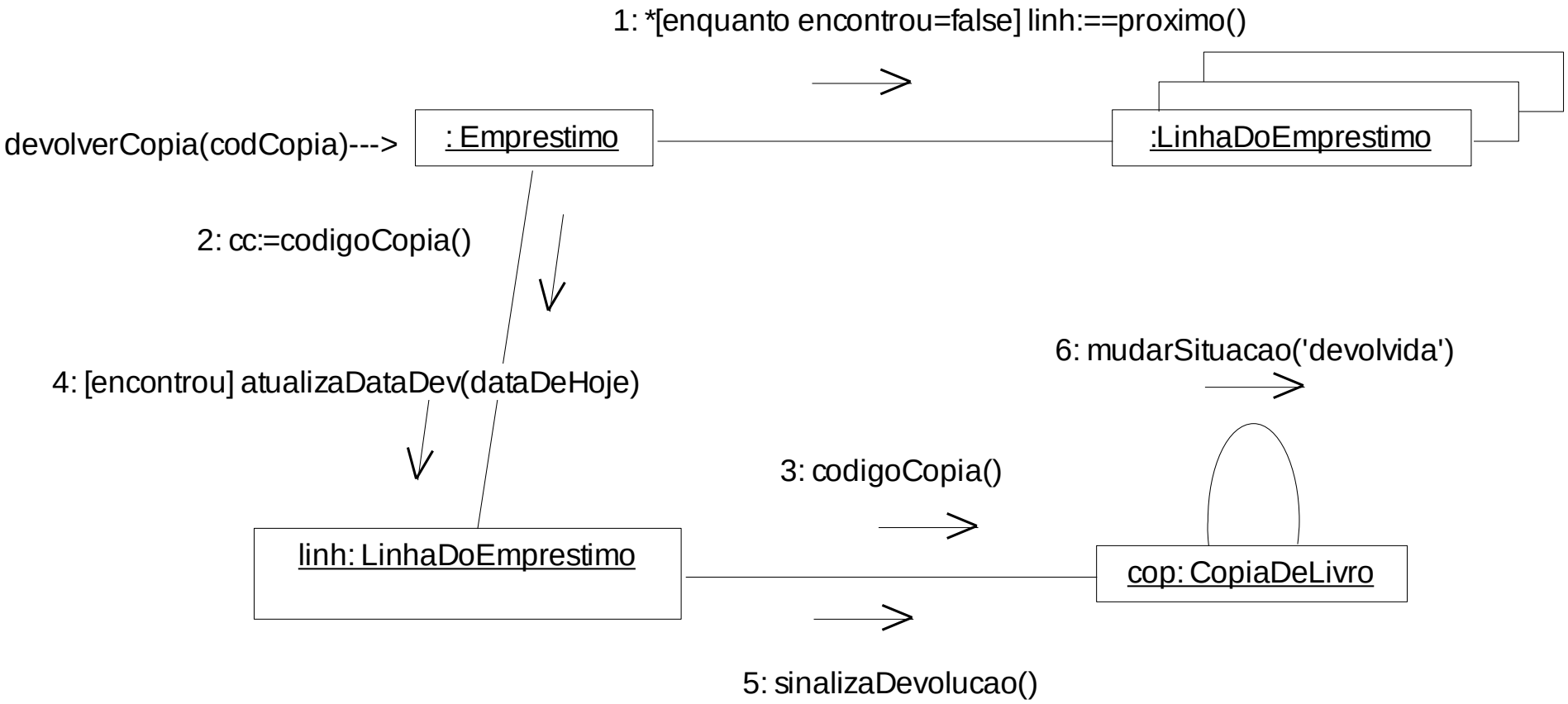
Inclusão de métodos



Este diagrama de comunicação implica nos seguintes métodos:

Emprestimo: adicionarCopia()

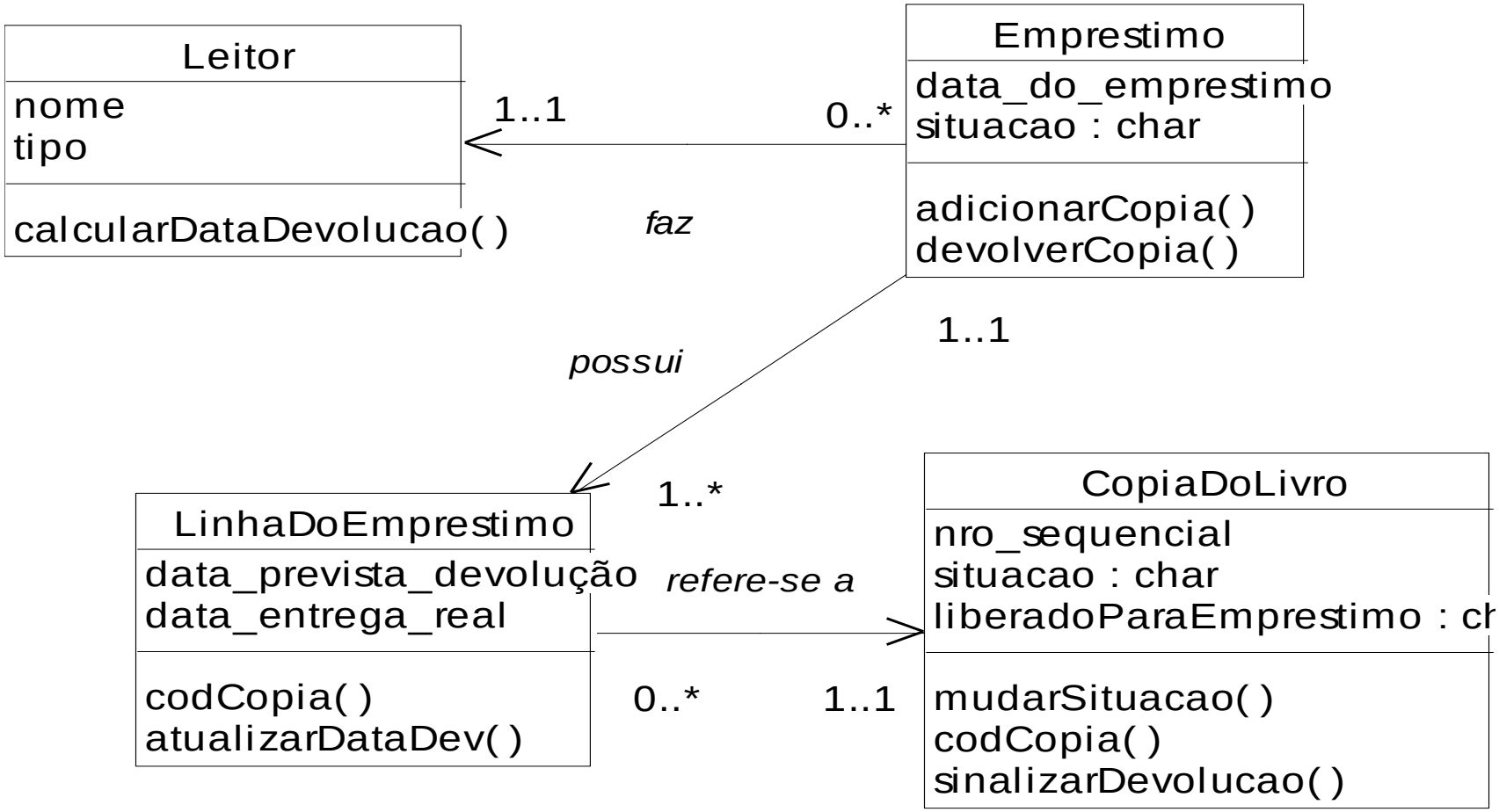
Leitor: calcularDataDevolucao()



Este diagrama de comunicação implica nos métodos:

Emprestimo: devolverCopia()
LinhaEmprestimo: codigoCopia()
LinhaEmprestimo: atualizaDataDev()
CopiaDeLivro: mudarSituacao()
CopiaDeLivro: codigoCopia()
CopiaDeLivro: sinalizaDevolucao()

Diagrama de Classes resultante



(com base apenas nos 2 diagramas de comunicação mostrados)

Atributos

- Pode-se acrescentar tipos de atributos, parâmetros e retornos de métodos, observando os diagramas de comunicação
- Atributos identificados durante o projeto podem ser incluídos
- se uma ferramenta CASE for utilizada para geração automática de código, os tipos detalhados são necessários
- se o diagrama for usado exclusivamente por desenvolvedores de software, o excesso de informação pode “poluir” o diagrama e dificultar seu entendimento

Observações

- Novas classes podem surgir nos diagramas de comunicação, portanto deve-se pensar em nomes para elas, bem como nas multiplicidades das associações correspondentes.

Referências

- C. Larman. Utilizando UML e Padrões, Editora Bookman, 3ª edição, 2008. Cap. 18 e 19, p. 227-257.