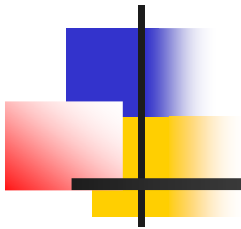


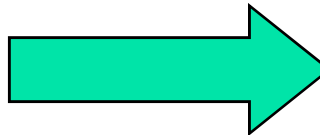
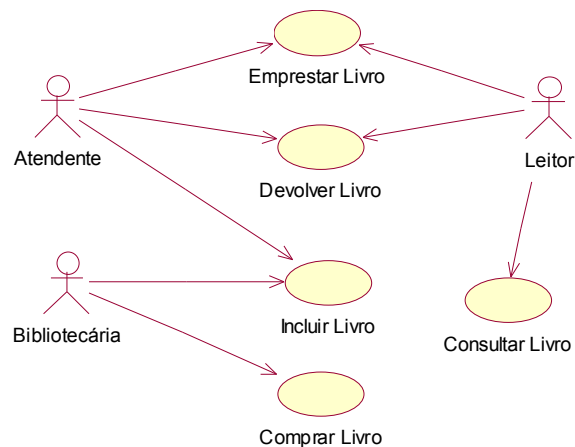
Visibilidade entre objetos e Diagramas de Classes



Profa Dra Rosana T. V. Braga

O que já foi visto até agora

Diagrama de Casos de Uso



Casos de Uso Completo Abstrato

Caso de Uso: Emprestar Livro

Ator Principal: Atendente

Interessados e Interesses:

- Atendente: deseja registrar que um ou mais livros estão em posse de um leitor, para controlar se a devolução será feita no tempo determinado.
- Leitor: deseja emprestar um ou mais livros, de forma rápida e segura.
- Bibliotecário: deseja controlar o uso dos livros, para que não se percam e para que sempre se saiba com que leitor estão no momento.

Pré-Condições: O Atendente é identificado e autenticado.

Garantia de Sucesso (Pós-Condições): Os dados do novo empréstimo estão armazenados no Sistema. Os livros emprestados possuem status "emprestado".

Cenário de Sucesso Principal:

1. O Leitor chega ao balcão de atendimento da biblioteca e diz ao atendente que deseja emprestar um ou mais livros da biblioteca.
2. O Atendente seleciona a opção para realizar um novo empréstimo.
3. O Atendente solicita ao leitor sua carteira de identificação, seja de estudante ou professor.
4. O Atendente informa ao sistema a identificação do leitor.
5. O Sistema exibe o nome do leitor e sua situação.
6. O Atendente solicita os livros a serem emprestados.
7. Para cada um deles, informa ao sistema o código de identificação do livro.
8. O Sistema informa a data de devolução de cada livro.
9. Se necessário, o Atendente desbloqueia os livros para que possam sair da biblioteca.
10. O Leitor sai com os livros.

Fluxos Alternativos:

- (1-8). A qualquer momento o Leitor informa ao Atendente que desistiu do empréstimo.
3. O Leitor informa ao Atendente que esqueceu a carteira de identificação.
 1. O Atendente faz uma busca pelo cadastro do Leitor e pede a ele alguma informação pessoal para garantir que ele é mesmo quem diz ser.
4. O Leitor está impedido de fazer empréstimo, por ter não estar apto.
 1. Cancelar a operação.
- 7a. O Livro não pode ser emprestado, pois está reservado para outro leitor.
 1. O Atendente informa ao Leitor que não poderá emprestar o livro e pergunta se deseja reservá-lo.
 2. Cancelar a operação (se for o único livro)
- 7b. O Livro não pode ser emprestado, pois é um livro reservado somente para consulta.
 1. Cancelar a operação (se for o único livro)

O que já foi visto até agora

Casos de Uso com substantivos e verbos sublinhados

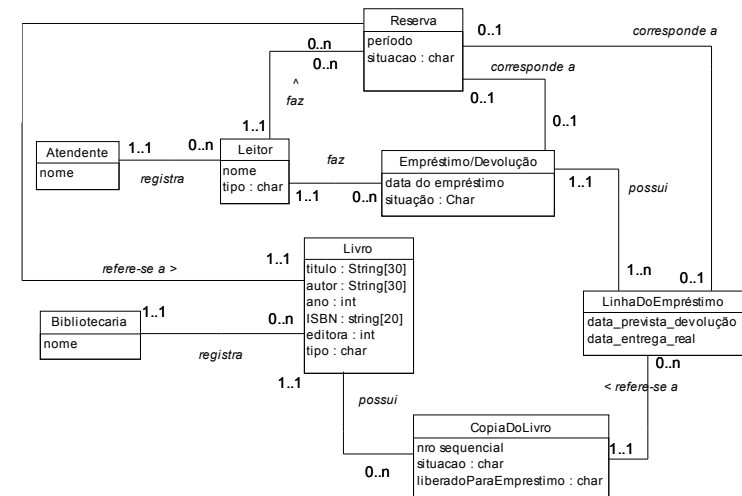
Caso de Uso 1

1. O Leitor chega ao balcão de atendimento da biblioteca e diz ao atendente que deseja emprestar um ou mais livros da biblioteca.
2. O Atendente seleciona a opção para adicionar um novo empréstimo.
3. O Atendente solicita ao leitor sua carteirinha, seja de estudante ou professor.
4. O Atendente informa ao sistema a identificação do leitor.
5. O Sistema exibe o nome do leitor e sua situação.
6. O Atendente solicita os livros a serem emprestados.
7. Para cada um deles, informa ao sistema o código de identificação do livro.
8. O Sistema informa a data de devolução de cada livro.
9. O Atendente desbloqueia os livros para que possam sair da biblioteca.
10. O Leitor sai com os livros.

Caso de Uso n

1. O Leitor chega ao balcão de atendimento da biblioteca e diz ao atendente que deseja emprestar um ou mais livros da biblioteca.
2. O Atendente seleciona a opção para adicionar um novo empréstimo.
3. O Atendente solicita ao leitor sua carteirinha, seja de estudante ou professor.
4. O Atendente informa ao sistema a identificação do leitor.
5. O Sistema exibe o nome do leitor e sua situação.
6. O Atendente solicita os livros a serem emprestados.
7. Para cada um deles, informa ao sistema o código de identificação do livro.
8. O Sistema informa a data de devolução de cada livro.
9. O Atendente desbloqueia os livros para que possam sair da biblioteca.
10. O Leitor sai com os livros.

Modelo Conceitual



O que já foi visto até agora

Modelo Conceitual
+
Casos de Uso

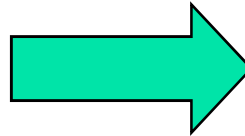
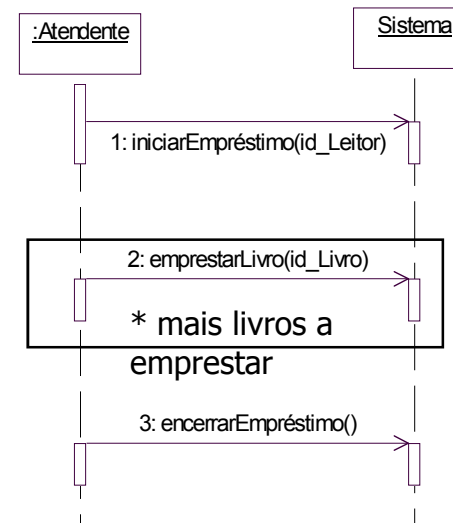
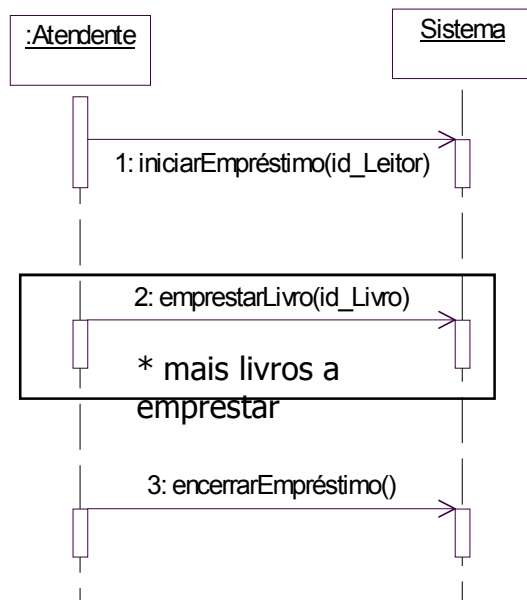


Diagrama de Seqüência do Sistema
(para cada caso de uso)



O que já foi visto até agora

Diagrama de Seqüência do Sistema
(para cada caso de uso)



Contrato da Operação
(para cada operação)

Operação: encerrarEmpréstimo()

Referências Cruzadas: Caso de uso: “Emprestar Livro”

Pré-Condições: Um leitor apto a emprestar livros já foi identificado; pelo menos um livro já foi identificado e está disponível para ser emprestado.

Pós-Condições: um novo empréstimo foi registrado; o novo empréstimo foi relacionado ao leitor já identificado na operação “iniciar o empréstimo”; a situação dos livros emprestados foi alterada para “emprestado”.

O que já foi visto até agora

Contrato da Operação (para cada operação)

Operação: encerrarEmpréstimo()

Referências Cruzadas: Caso de uso: “Emprestar Livro”

Pré-Condições: Um leitor apto a emprestar livros já foi identificado; pelo menos um livro já foi identificado e está disponível para ser emprestado.

Pós-Condições: um novo empréstimo foi registrado; o novo empréstimo foi relacionado ao leitor já identificado na operação “iniciar o empréstimo”; a situação dos livros emprestados foi alterada para “emprestado”.

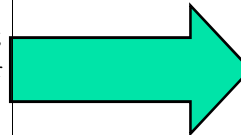
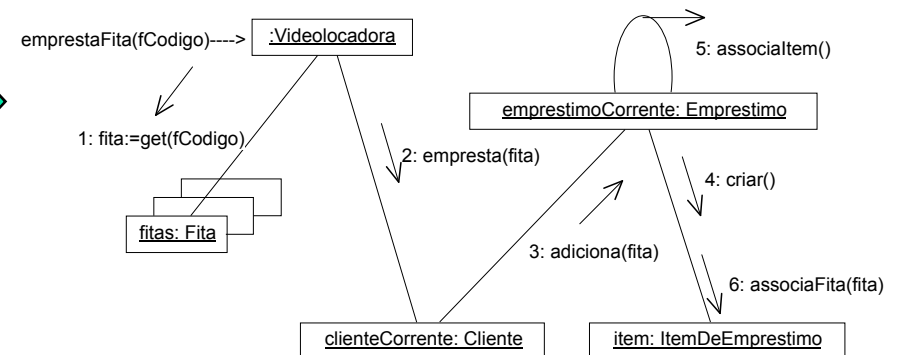


Diagrama de Colaboração (para cada operação)



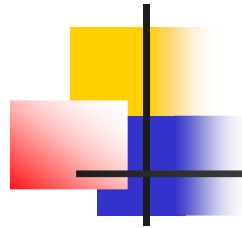


Visibilidad entre Objetos



Visibilidade entre Objetos

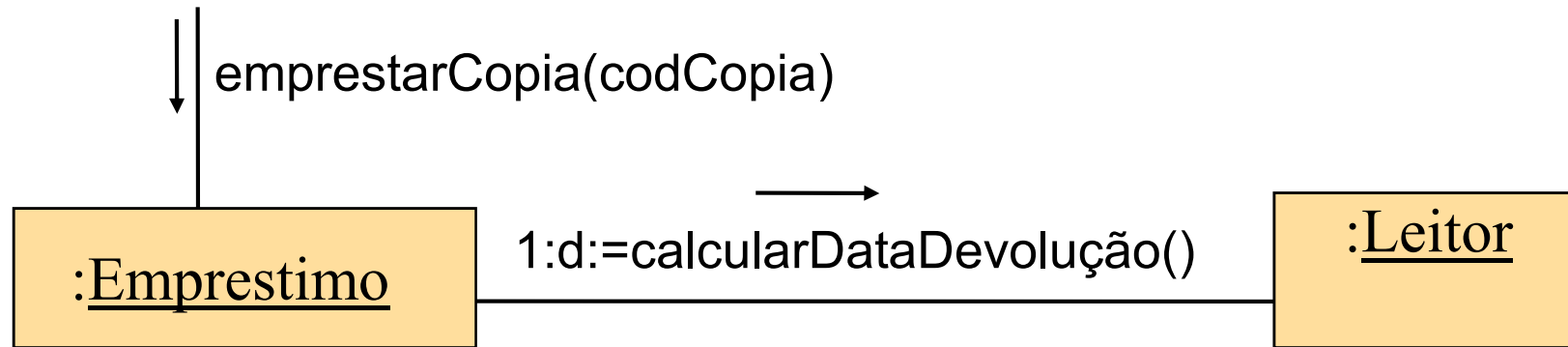
- Visibilidade: capacidade de um objeto ver ou fazer referência a outro
- Para que um objeto A envie uma mensagem para o objeto B, é necessário que B seja visível para A
- Tipos de visibilidade
 - por atributo: B é um atributo de A
 - por parâmetro: B é um parâmetro de um método de A
 - localmente declarada: B é declarado como um objeto local em um método de A
 - global: B é, de alguma forma, globalmente visível.



Visibilidade por atributo

- Persiste por muito tempo
- É a forma mais comum
- Geralmente se deve às associações existentes no modelo conceitual
- Ex: Empréstimo tem um atributo para poder enviar mensagens ao Leitor que efetuou o empréstimo.

Exemplo

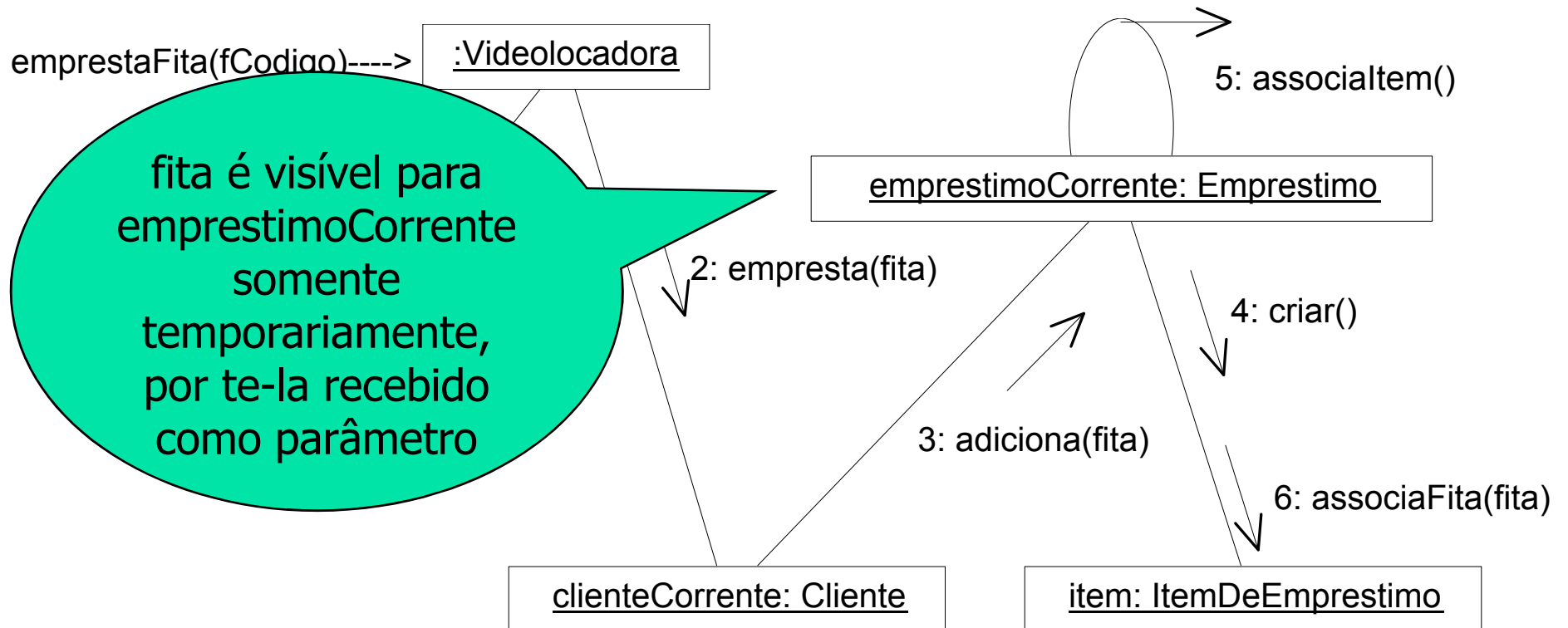


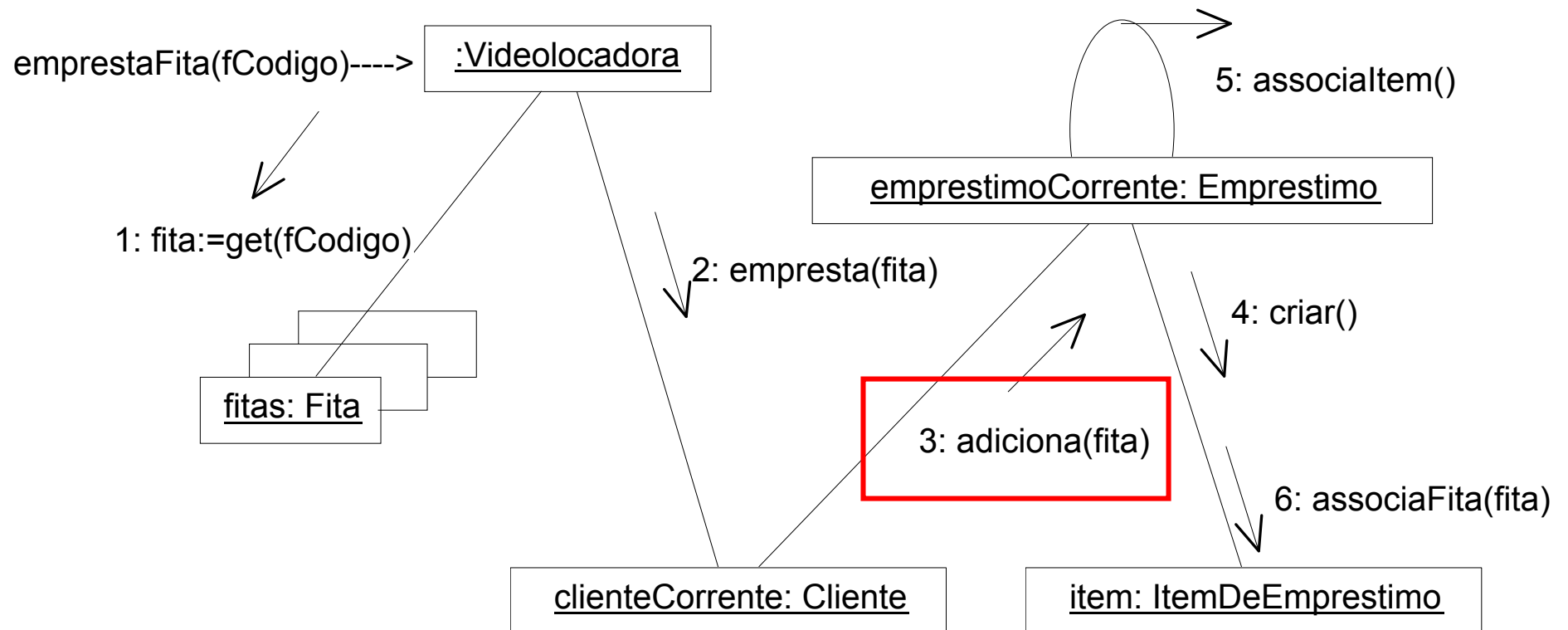
```
class Emprestimo
{
    ...
    private Leitor leitor;
    ...
    public void emprestarCopia(...);
    ...
}
```

```
public void emprestarCopia(int codCopia)
{
    LinhaDoEmprestimo linh;
    ...
    d = leitor.calcularDataDevolucao();
    ...
}
```

Visibilidade por parâmetro

- É relativamente temporária, persiste enquanto persistir o método.





```
public void adiciona(Fita fita)
{
    ...
    ItemDeEmprestimo item = new ItemDeEmprestimo();
    this.associaItem();
    item.associaFita(fita);
    ...
}
```


fita é passada como parâmetro para que *emprestimoCorrente* tenha visibilidade de *Fita*.



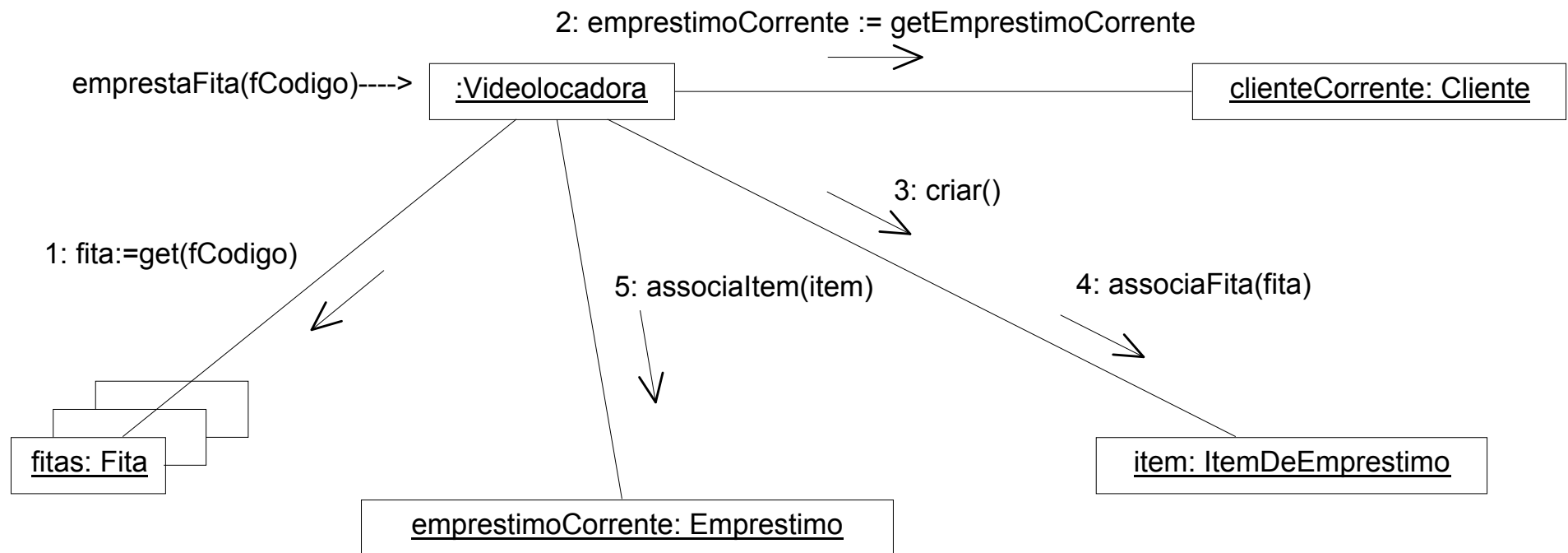
Visibilidade localmente declarada

- Visibilidade relativamente temporária
- Duas formas:
 - criar uma nova instância local e atribuí-la a uma variável local
 - atribuir o objeto retornado pela invocação de um método a uma variável local.

criação de instância local

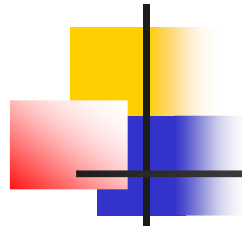


```
public void adiciona(Fita fita)
{
...
ItemDeEmprestimo item = new ItemDeEmprestimo();
this.associaItem();
item.associaFita(fita);
...
}
```



atribuição de objeto retornado à variável local

```
public void emprestaFita(int fCodigo)
{
    Emprestimo emprestimoCorrente;
    ...
    emprestimoCorrente = clienteCorrente.getEmprestimoCorrente();
    ...
}
```



Visibilidade Global

- Menos comum
- Relativamente permanente (persiste enquanto A ou B existirem)
- Forma óbvia e menos desejável: atribuir uma instância de objeto a uma variável global.

Notação de Visibilidade em UML

<<associação>> é usada para visibilidade de atributo

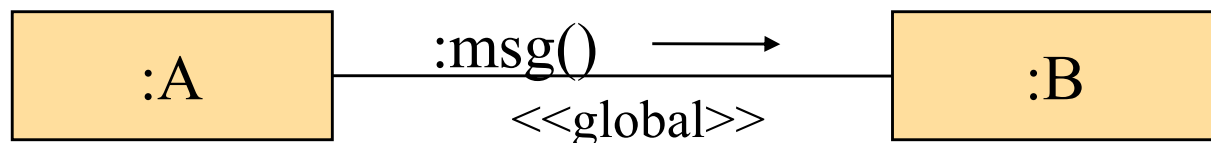
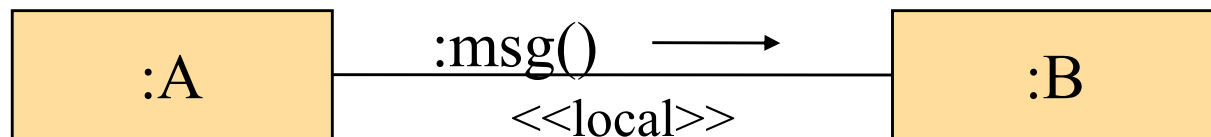
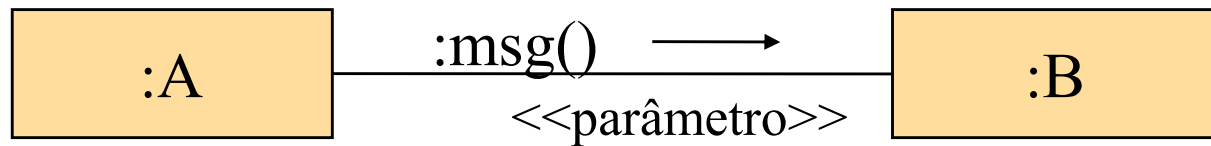
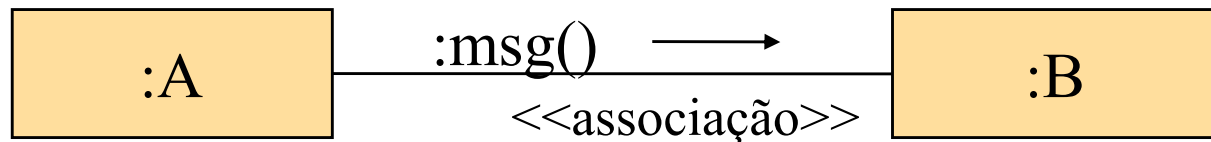


Diagrama de Classes de Projeto

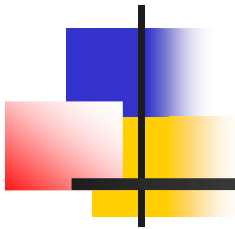
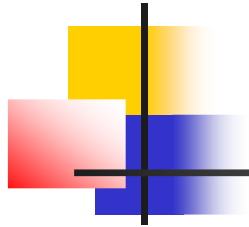


Diagrama de Classes de Projeto



- O Diagrama de Classes de Projeto apresenta especificações para classes de software e interfaces (ex: interfaces Java) de uma aplicação
- Informação típica:
 - classes, associações e atributos
 - interfaces, com operações e constantes
 - métodos
 - tipos dos atributos
 - navegabilidade
 - dependências



Definição (cont.)

- 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



Definição (cont.)

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



Classes que aparecem nos 2 diagr. de colaboração

Leitor
nome
tipo

Emprestimo
data_do_emprestimo
situacao : char

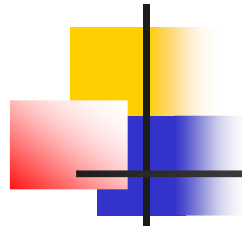
LinhaDoEmprestimo
data_prevista_ devolução

CopiaDoLivro
nro_sequencial
situacao : char
liberadoParaEmprestimo :



Associações e Navegabilidade

- Associações e navegabilidade entre classes são indicadas pelos diagramas de colaboração
 - Navegabilidade indica possibilidade de navegação unidirecional por meio de uma associação entre classes
 - geralmente implica visibilidade por atributos
- A multiplicidade e os nomes das associações são retirados do Modelo Conceitual
- Notação: seta contínua



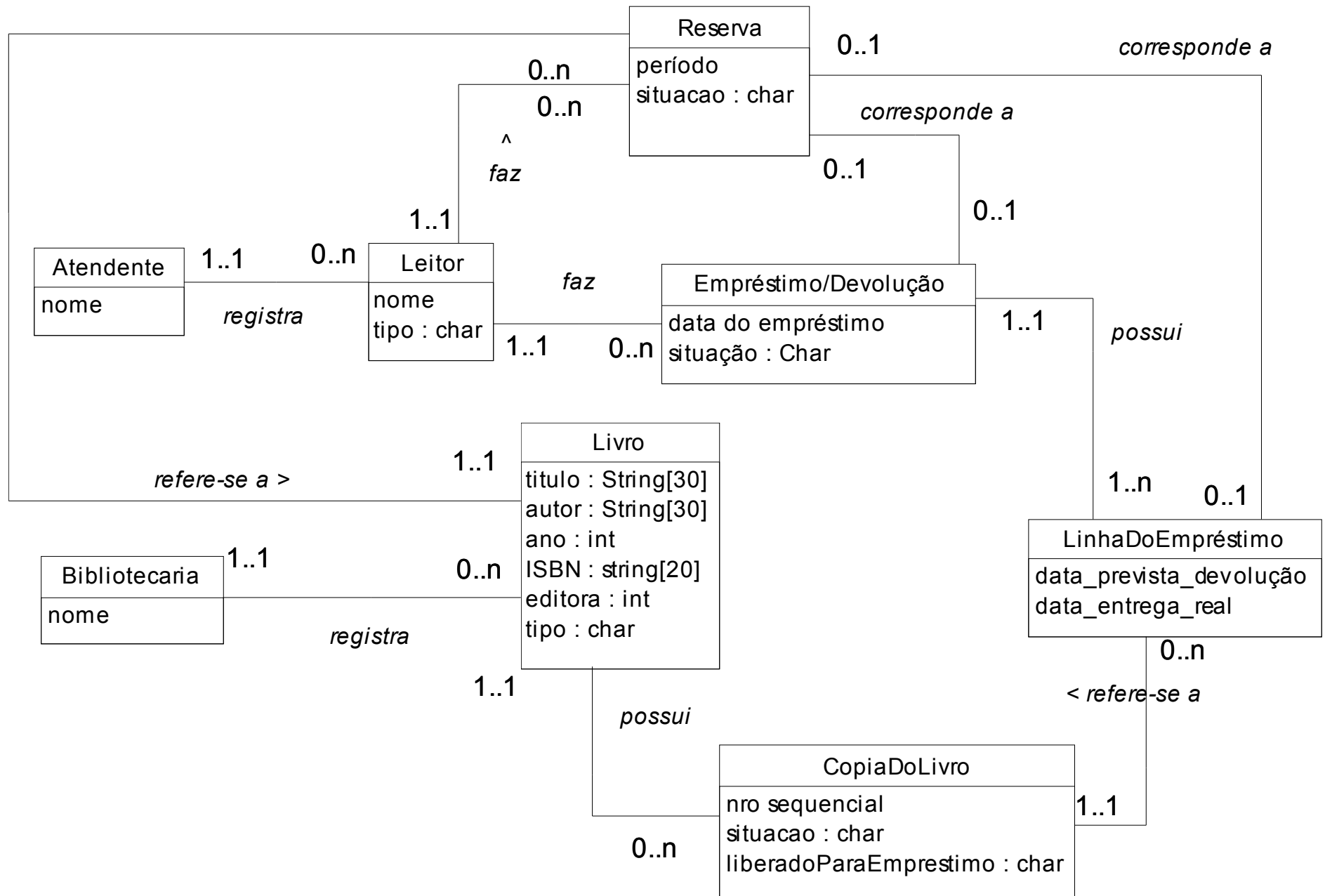
Associações e Navegabilidade

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



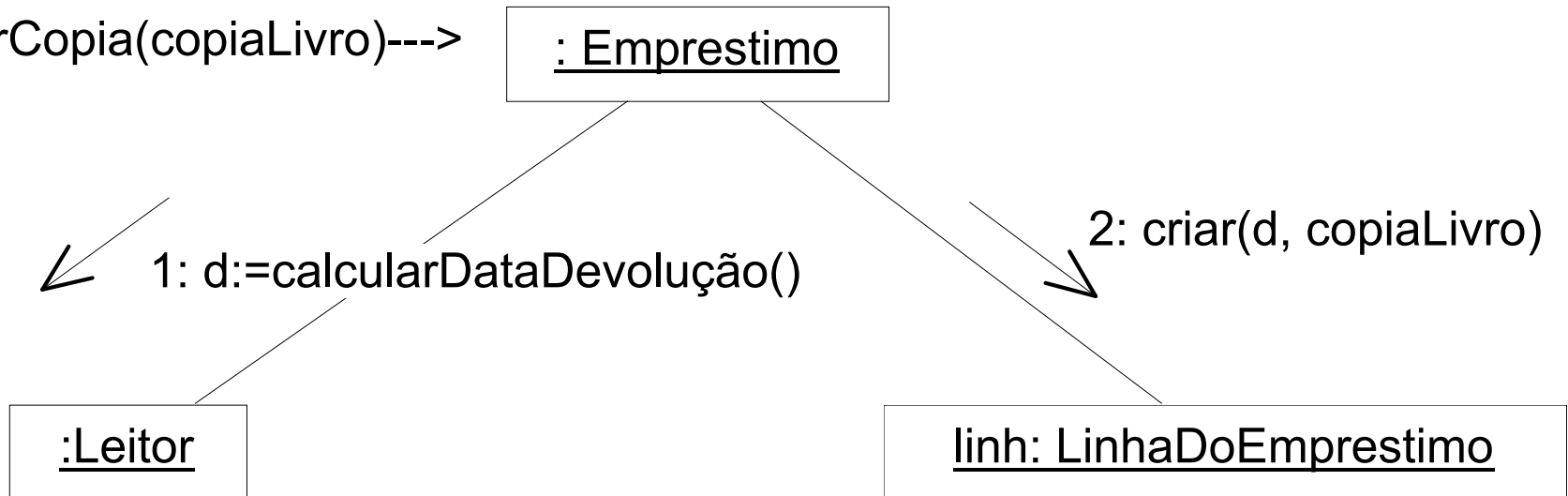
Como determinar a navegabilidade?

- 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

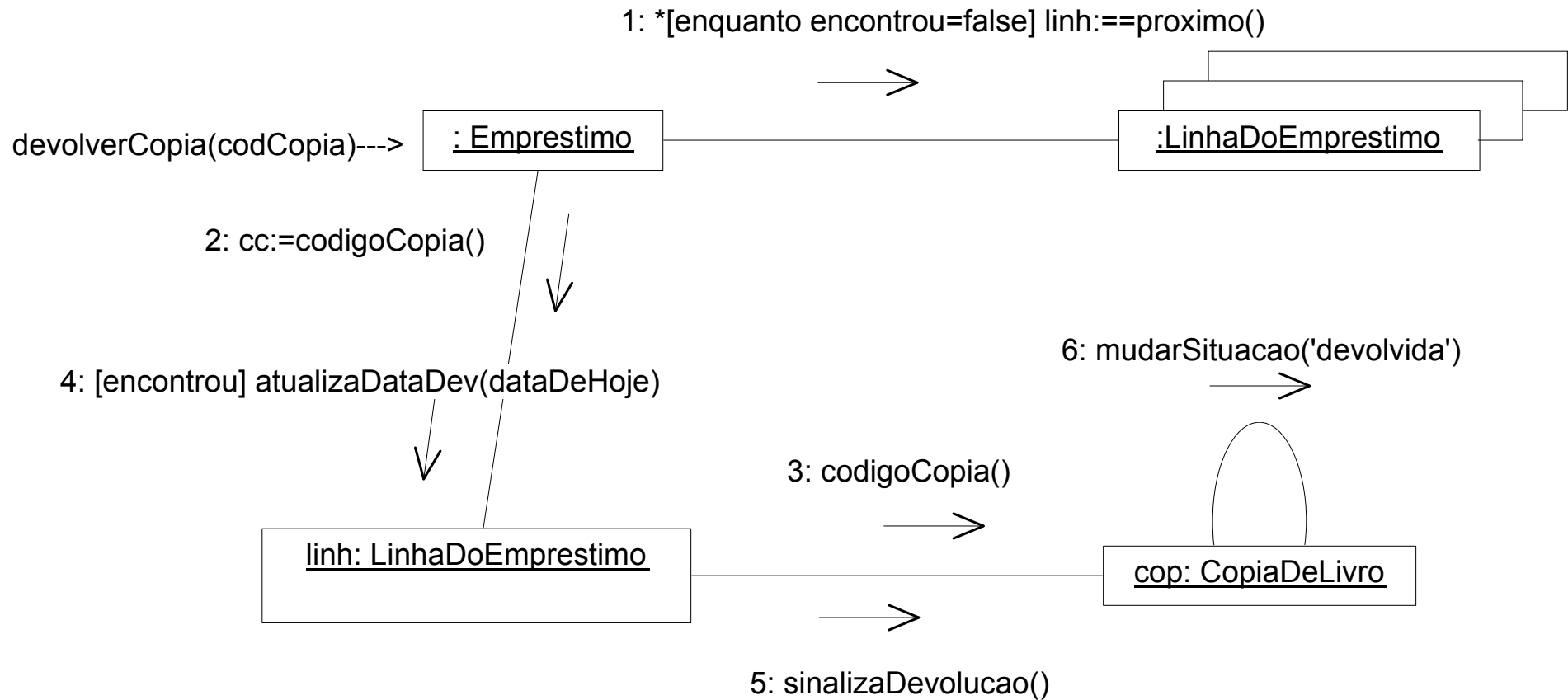
adicionarCopia(copiaLivro)--->



este diagrama de colaboração implica nas navegabilidades:

`mprestimo` → `Leitor`

`mprestimo` → `LinhaDoEmprestimo`

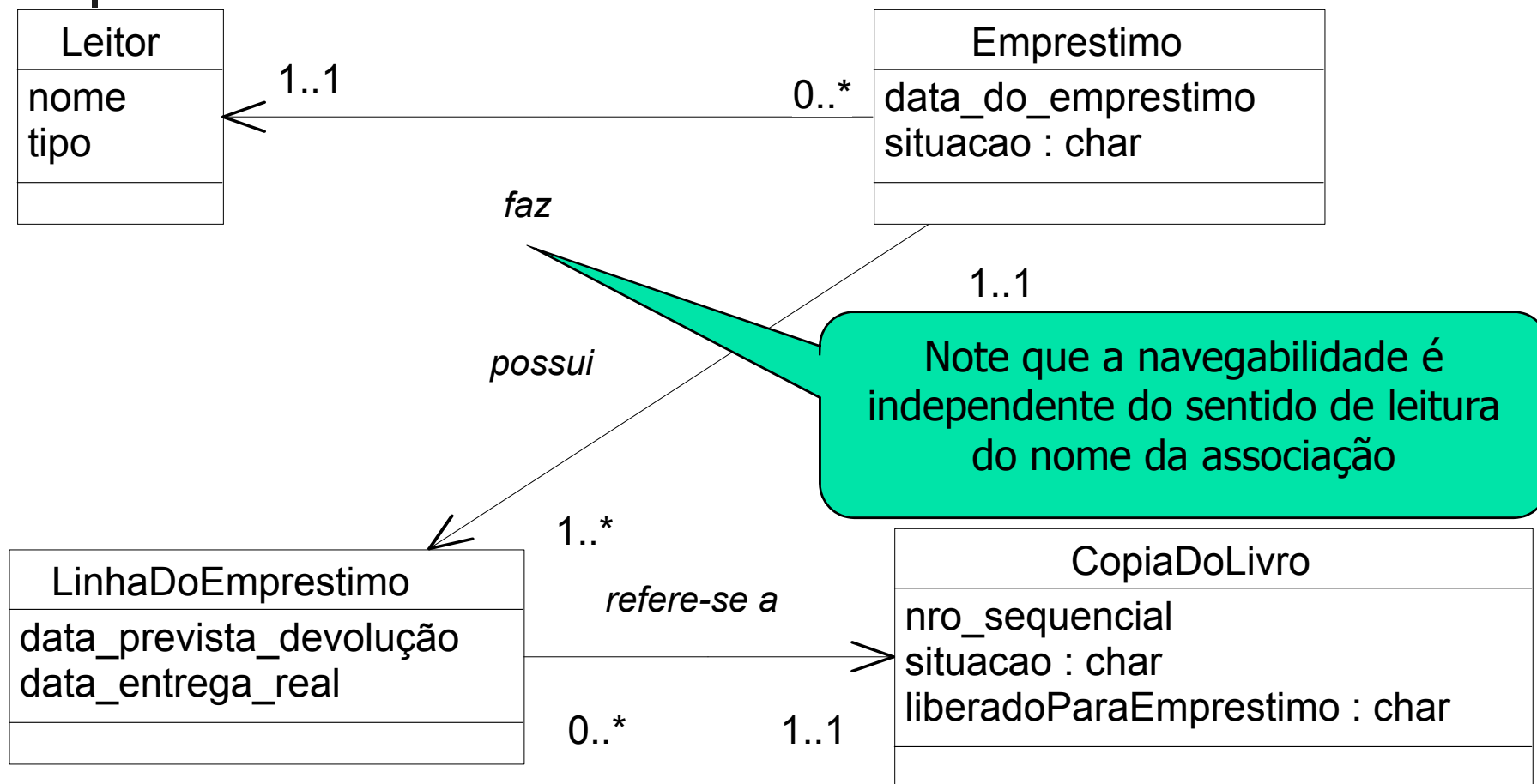


Este diagrama de colaboração implica nas navegabilidades:

Emprestimo → LinhaDoEmprestimo

LinhaEmprestimo → CopiaDoLivro

Diagrama de Classes com navegabilidade



(com base apenas nos 2 diagr. de colaboração mostrados)



Como incluir os métodos nas classes?

- Operações são incluídas nas classes controladoras
- 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($Par_1, Par_2, \dots Par_n$)

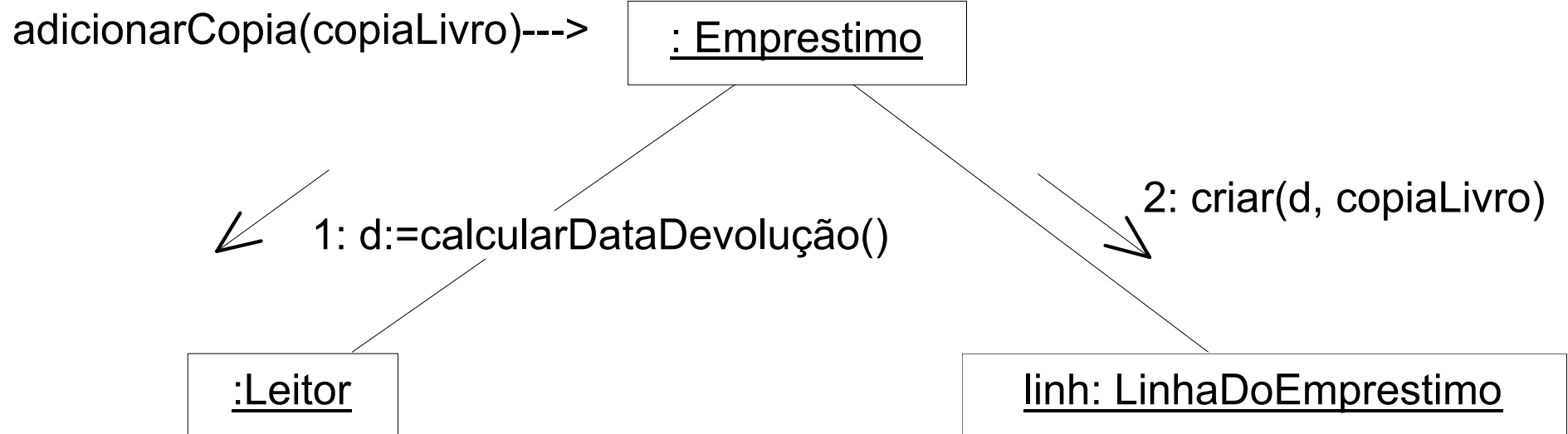


Como incluir os métodos nas classes?

- Não incluir:
 - Métodos enviados à coleções (dependem da implementação)
 - Método criar (linguagem OO provê o criador)
 - métodos de acesso a atributos, por exemplo, setNome, getNome, etc. → assume-se que cada atributo tem necessariamente esses métodos



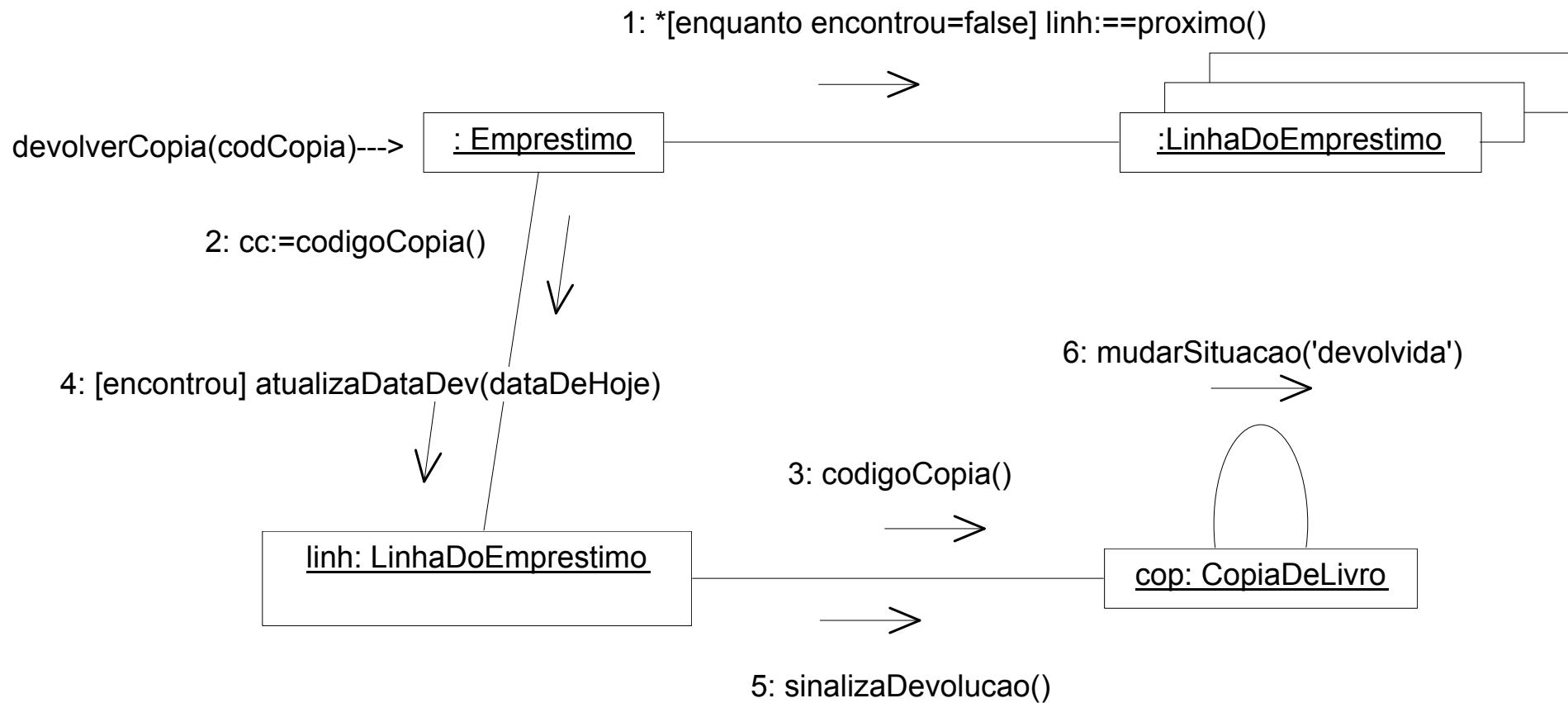
Inclusão de métodos



este diagrama de colaboração implica nos seguintes métodos:

mprestimo → adicionarCopia()

leitor → calcularDataDevolucao()



Este diagrama de colaboração implica nos métodos:

Emprestimo → devolverCopia()

LinhaEmprestimo → codCopia()

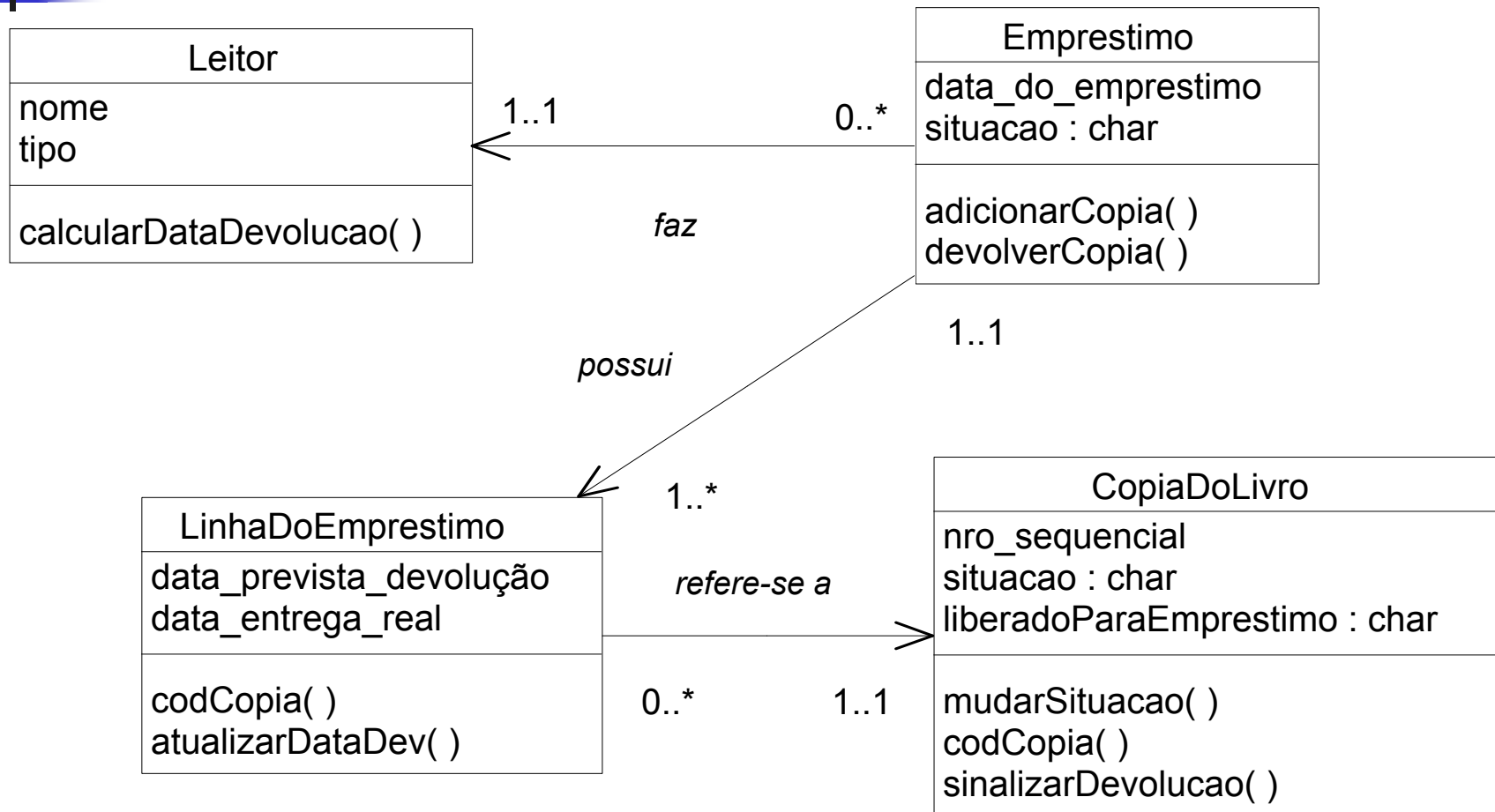
LinhaEmprestimo → atualizaDataDev()

CopiaDeLivro → mudarSituacao()

CopiaDeLivro → codCopia()

CopiaDeLivro → sinalizaDevolucao()

Diagrama de Classes resultante



em base apenas nos 2 diagr. de colaboração mostrados)



Atributos

- Pode-se acrescentar tipos de atributos, parâmetros e retornos de métodos, observando os diagr. de colaboração
- 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

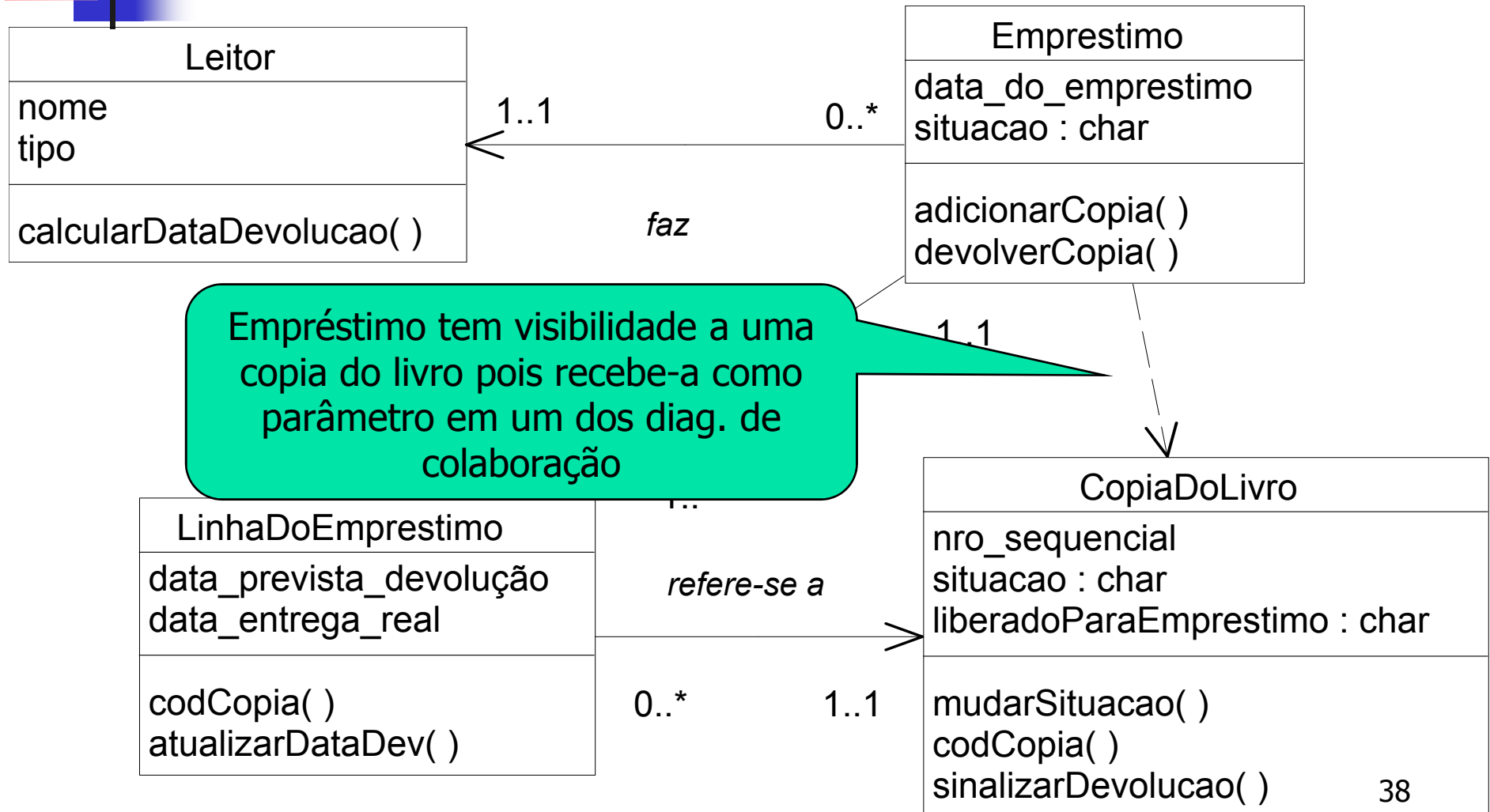
- Embora uma visibilidade por atributo venha a ser implementada posteriormente como um atributo na classe origem, isso não deve ser mostrado no diagrama de classes
- Novas classes podem surgir nos diagramas de colaboração, portanto deve-se pensar em nomes para elas, bem como nas multiplicidades das associações correspondentes.



Relacionamento de Dependência

- No Diagrama de Classes, o relacionamento de dependência representa a visibilidade entre classes que não é implementada por atributo
 - visibilidade por parâmetro
 - visibilidade local ou global
- Um objeto de uma classe A tem conhecimento (enxerga) um objeto da classe B
- Notação: seta tracejada

Exemplo: Dependência



Alguns detalhes de notação para métodos e atributos - UML

