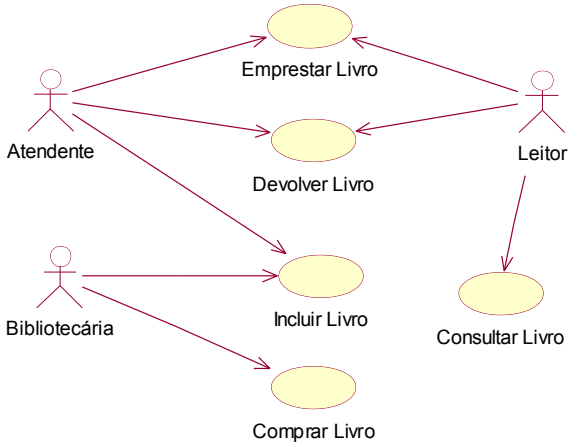
Do Projeto à Codificação



SSC-526 – Análise e Projeto Orientados a Objeto
Profa. Dra. Elisa Yumi Nakagawa
2º semestre de 2017

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

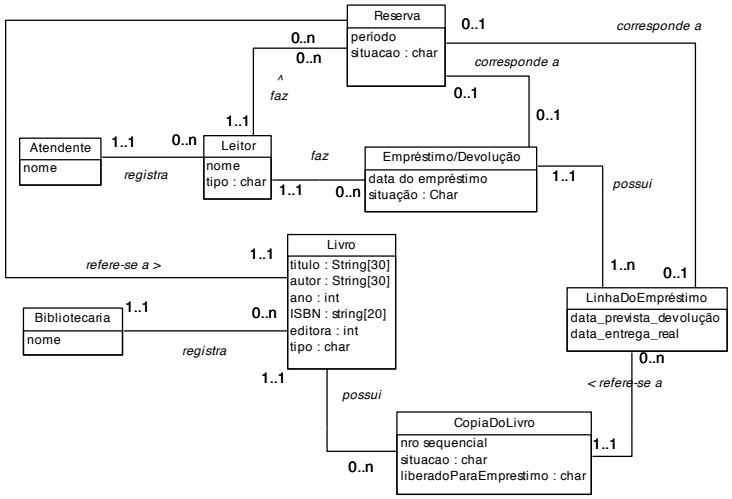
Caso de Uso 1

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.

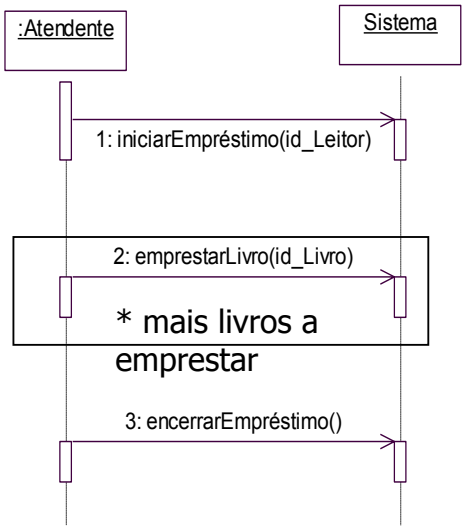
Modelo Conceitual



O que já foi visto até agora

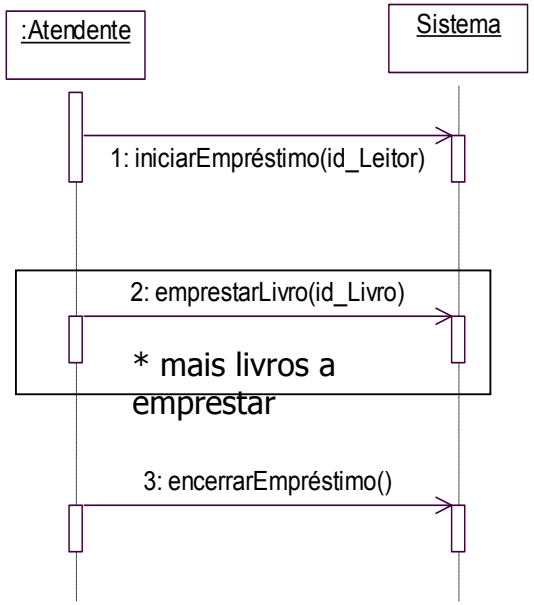
Modelo Conceitual
+
Casos de Uso

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)

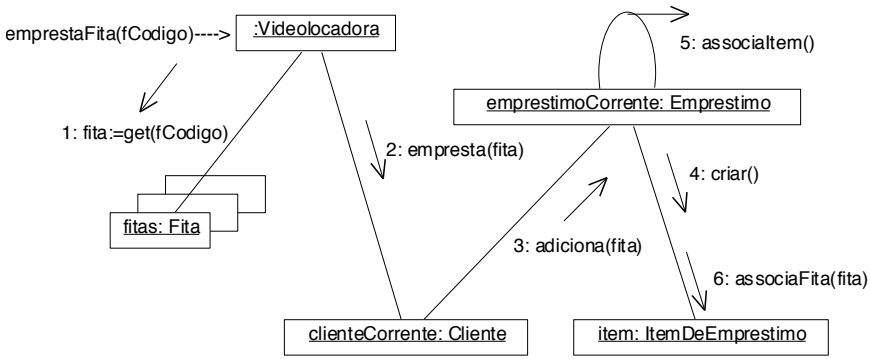
Operação: encerrarEmpréstimo()

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.

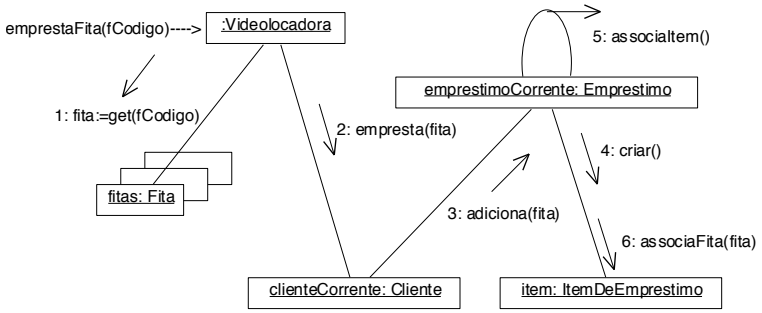
Pós-Condições: um novo empréstimo foi registrado; o novo empréstimo foi relacionado ao leitor já identificado na operação "iniciar o empréstimo"; a situação dos livros emprestados foi alterada para "emprestado".

Diagrama de Comunicação (para cada operação)

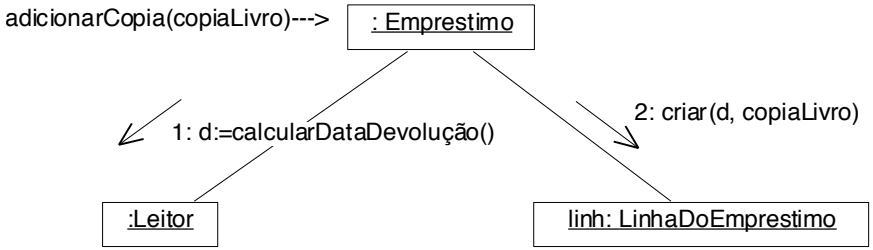


O que já foi visto até agora

Diagramas de Comunicação (para todas as operações)

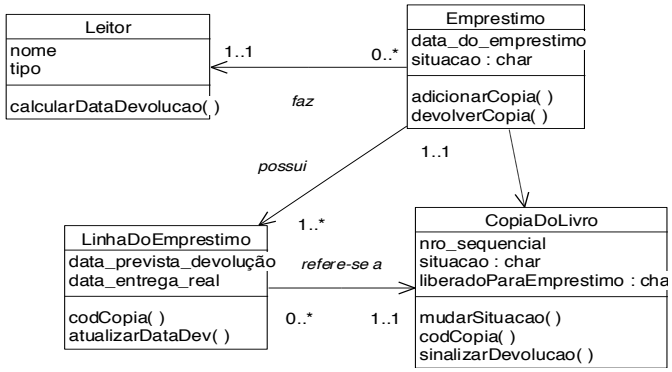


:Videolocadora
5: associaItem()
emprestimoCorrente: Emprestimo
1: fita:=get(fCodigo)
2: empresta(fita)
4: criar()
fitas: Fita
3: adiciona(fita)
6: associaFita(fita)
clienteCorrente: Cliente
item: ItemDeEmprestimo

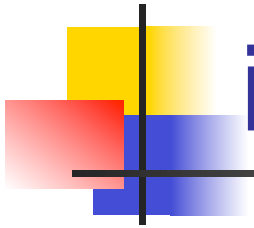


: Emprestimo
1: d:=calcularDataDevolução()
2: criar(d, copiaLivro)
:Leitor
linh: LinhaDoEmprestimo

Diagrama de Classes de Projeto

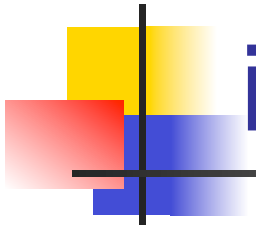


Como mapear o projeto para a implementação?

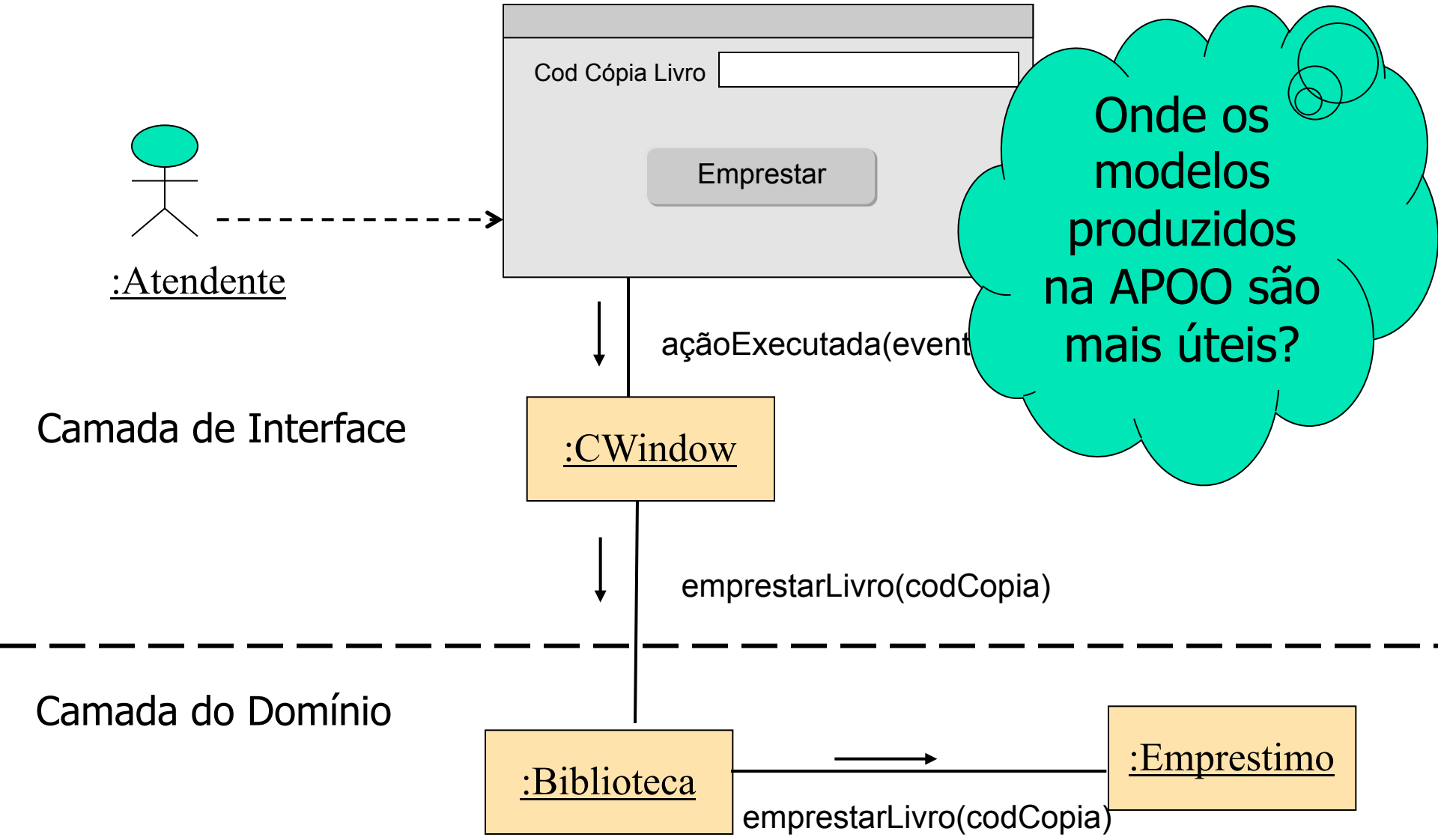


- Resultados obtidos no projeto são o ponto de partida, mas muito trabalho ainda tem que ser feito
- Muitas alterações podem ocorrer e problemas podem surgir e precisam ser solucionados
- Prepare-se: pode haver mudanças e desvios de projeto!!!

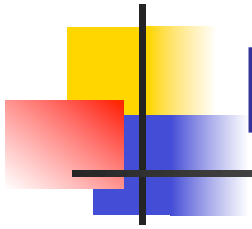
Como mapear o projeto para a implementação?



- Protótipos e código exploratório pode ter sido produzido na fase de projeto
- Ferramentas CASE (geração semi-automática do código) podem ajudar
- Código a ser escrito
  - Classes + Métodos
  - Interfaces
- Linguagem a ser usada como exemplo: JAVA



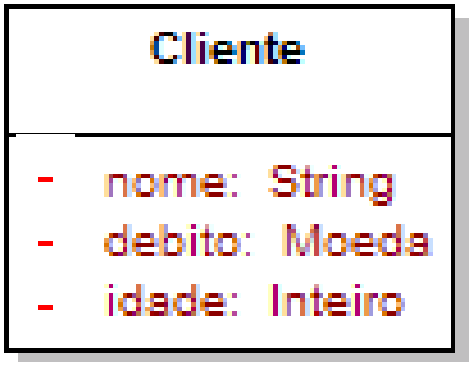
Resposta: Camada de domínio!!!



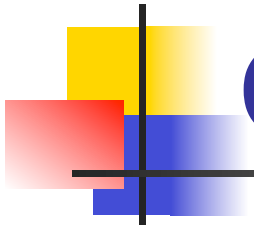
Definição de Classes

- Uma classe de programa deve ser criada para cada classe do Diagrama de Classes de Projeto.
- Método Criar → gera construtores em Java
- Tipos de atributos→ podem ser adotados tipos nativos da linguagem ou serem criados tipos a partir dos tipos nativos.
- Definição e assinaturas dos métodos

Classe em Java




```
class Cliente {
        private String nome;
        private Float debito;
        private Integer idade;
}
```



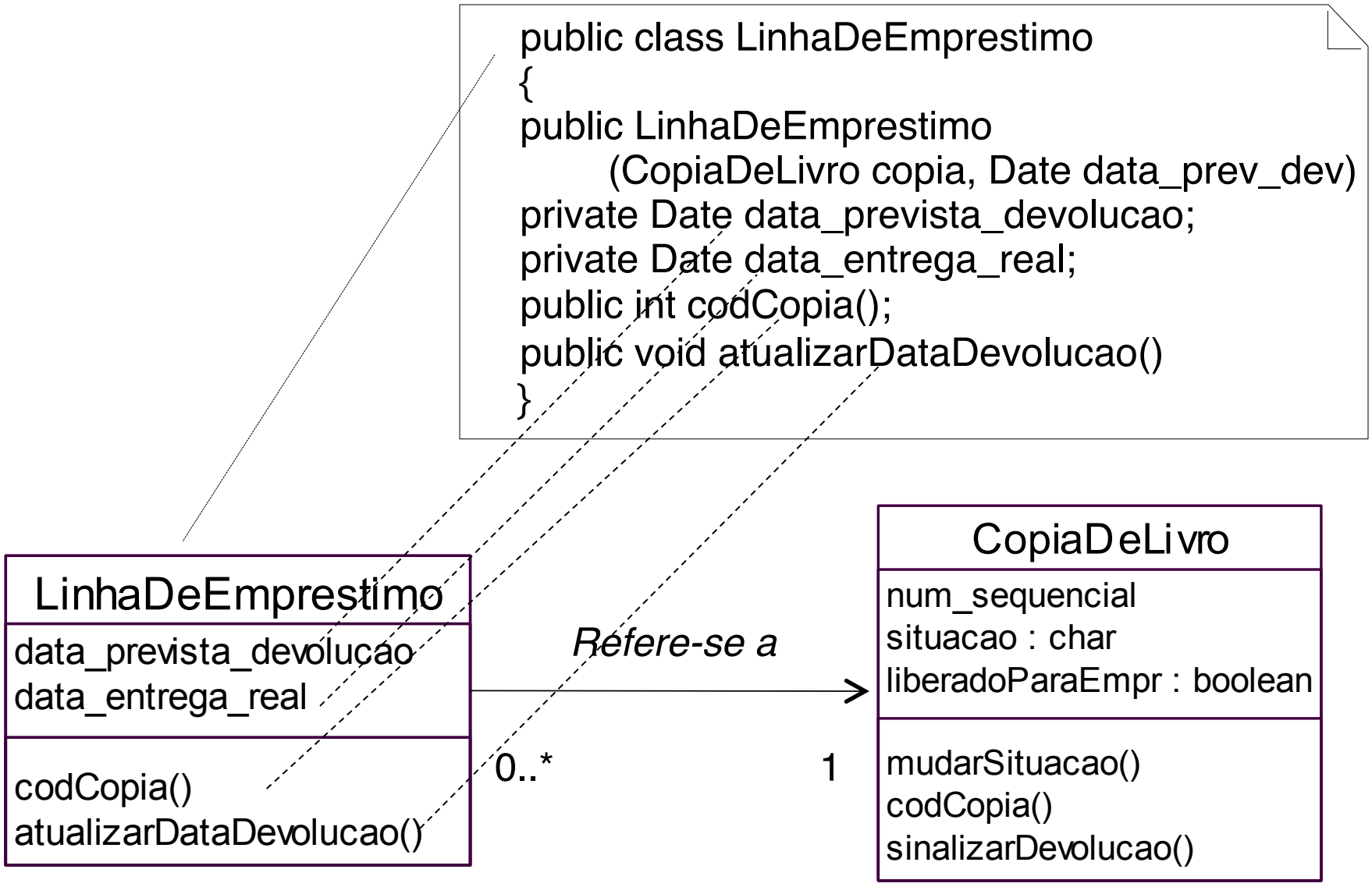
Classe em Java

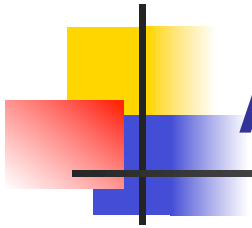
```
class Cliente {
        private String nome;
        private Float debito;
        private Integer idade;

        public void setNome(String nome) { this.nome = nome; }
        public void setDebito(Float debito) { this.debito = debito; }
        public void setIdade(Integer idade) { this.idade = idade; }

        public String getNome() { return nome; }
        public Float getDebito() { return debito; }
        public Integer getIdade() { return idade; }
}
```

Atributos comuns e assinatura de métodos

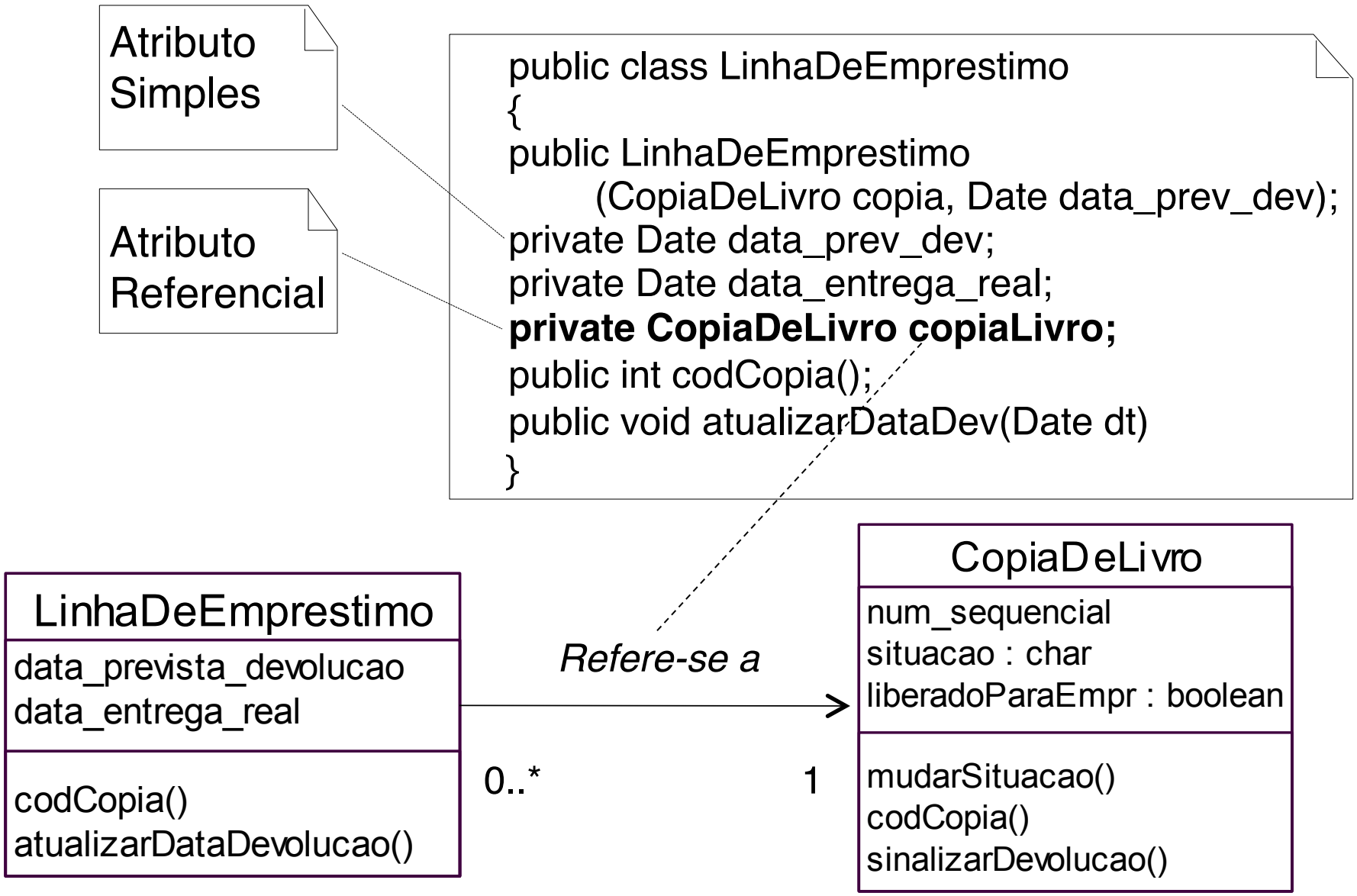


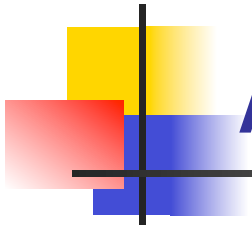


Atributos referenciais

- São atributos que referenciam um outro objeto complexo e não um tipo primitivo
- São sugeridos pelas associações e pela navegabilidade no diagrama de classes de projeto
- No diagrama de classes, os atributos referenciais estão normalmente implícitos (ao invés de explícitos como os demais atributos).

Atributo referencial





Associação para 1

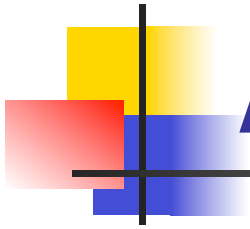
- Como associação é estritamente para 1
  - não é possível destruir a associação
  - método para destruir a associação não deve ser implementado.
- Como a associação para 1 é obrigatória para o objeto na origem
  - método criador da classe deve ter como parâmetro o elemento a ser associado.

Associação para 1




```
class Emprestimo {
        private ItemDeEmprestimo item;

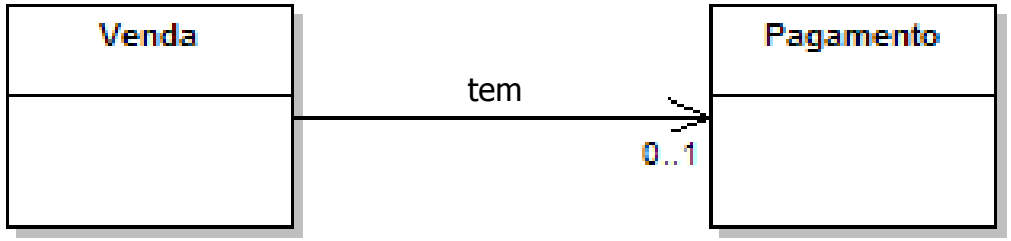
        public Emprestimo(ItemDeEmprestimo item) {
                this.associaItem(item)
        }
        public void associaItem(ItemDeEmprestimo item) {
                this.item = item;
        }
        public ItemDeEmprestimo getItem() {
                return item;
        }
}
```



Associação para 0..1

- É possível destruir a associação
  - deve ser implementado o método correspondente.
- Não é necessário passar um objeto como parâmetro para o método criador, pois a associação para 0..1 não é obrigatória.

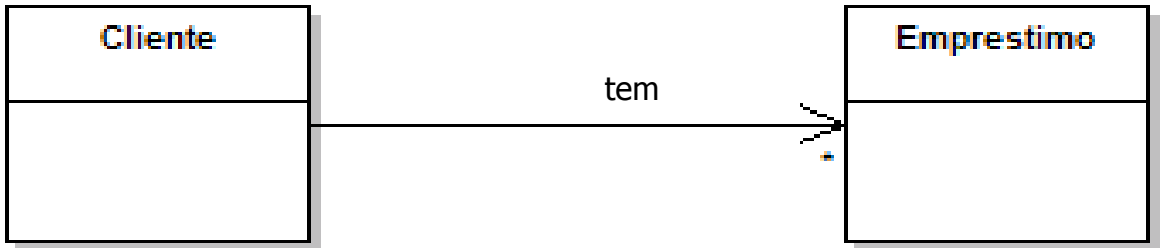
Associação para 0..1




```
class Venda {
        private Pagamento pagamento;

        public Venda() { }
        public void associaPagamento(Pagamento pagamento) {
                this.pagamento = pagamento;
        }
        public void desassociaPagamento() {
                this.pagamento = null;
        }
        public Pagamento getPagamento() {
                return pagamento;
        }
}
```

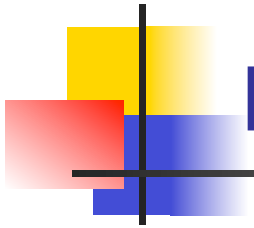
Associação para *



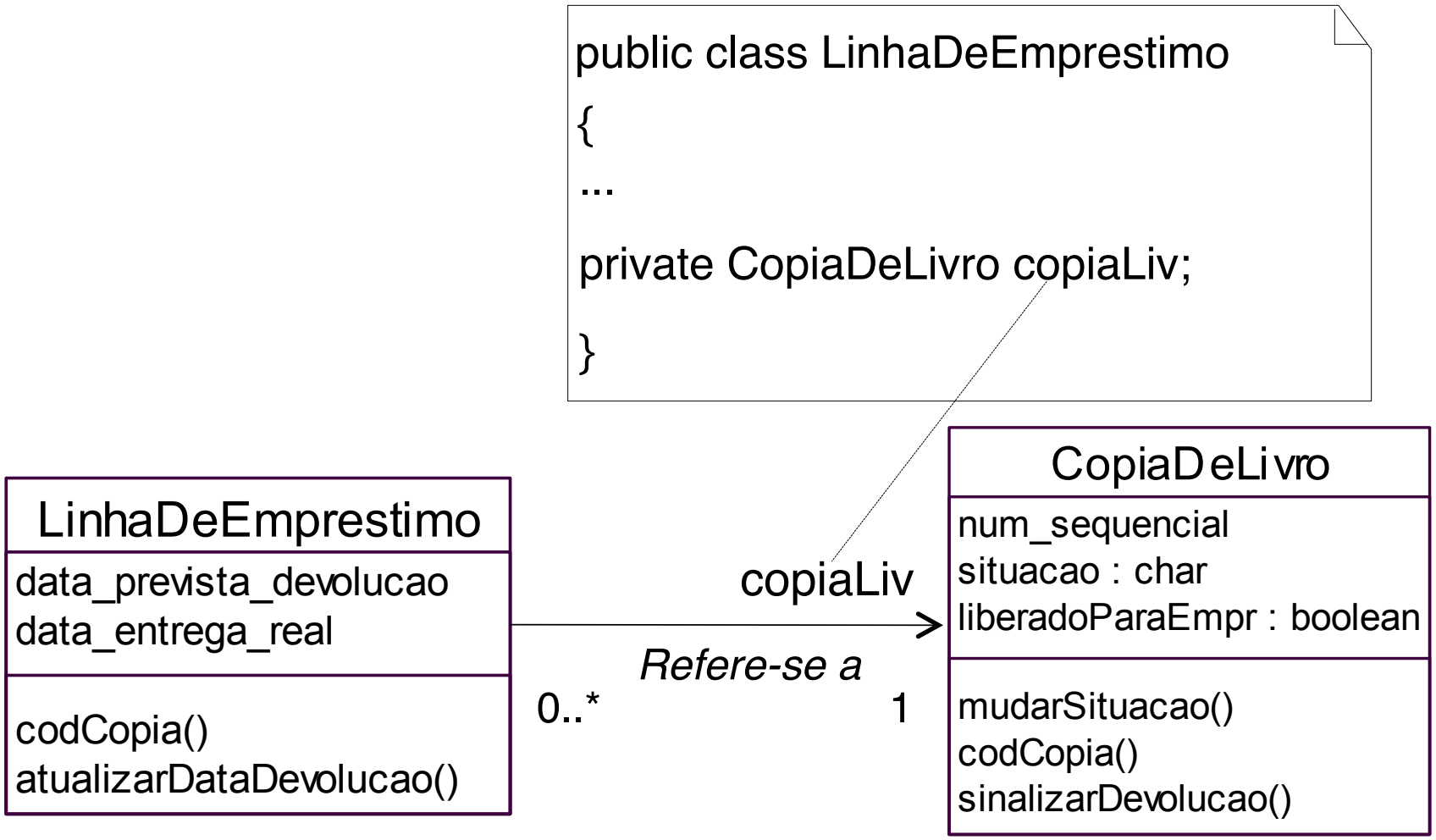

```
class Cliente {
        private Set emprestimos = new HashSet();

        public Cliente () { }
        public void adicionaEmprestimo(Emprestimo emprestimo) {
                this.emprestimos.add(emprestimo);
        }
        public void removeEmprestimo(Emprestimo emprestimo) {
                this.emprestimos.remove(emprestimo);
        }
        public Set getEmprestimos () {
                return Collections.unmodifiableSet(emprestimos);
        }
}
```

Atributos Referenciais e nomes de Papéis

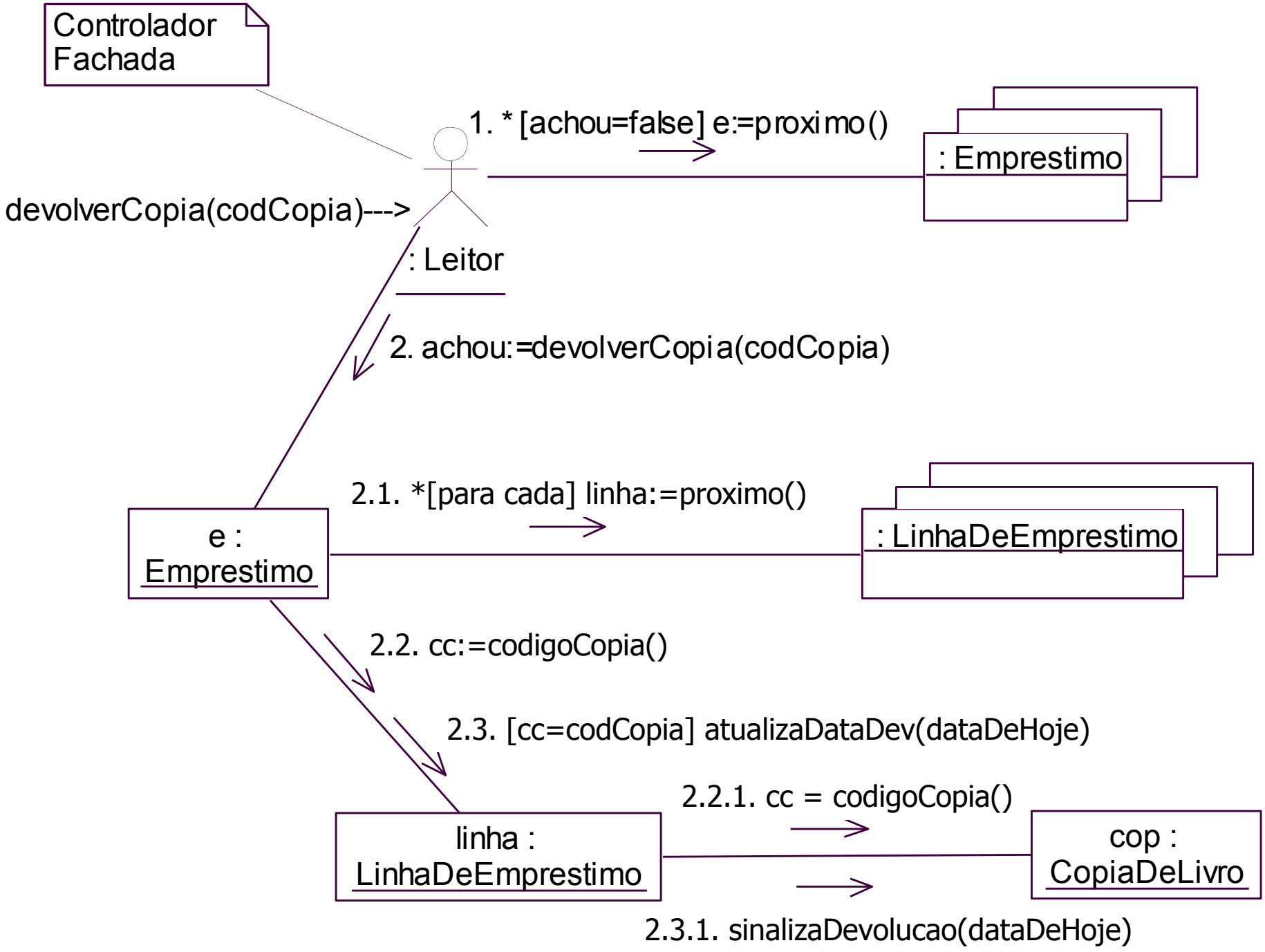


- O nome de papel identifica o papel da classe na associação e fornece frequentemente algum contexto semântico sobre a sua natureza.
- Se houver um nome de papel no diagrama de classes de projeto, utilize-o como base para o nome do atributo referencial durante a geração de código.


public class LinhaDeEmprestimo
{
...
private CopiaDeLivro copiaLiv;
}
LinhaDeEmprestimo
data_prevista_devolucao
data_entrega_real
codCopia()
atualizarDataDevolucao()
copiaLiv
Refere-se a
0..*
1
CopiaDeLivro
num_sequencial
situacao : char
liberadoParaEmpr : boolean
mudarSituacao()
codCopia()
sinalizarDevolucao()

Criar métodos a partir dos diagramas de comunicação

- Sequência de mensagens de um diagrama de comunicação é traduzida para uma série de comandos de programação na definição do método.
- Métodos de acesso (set e get) não serão ilustrados, por simplicidade, mas devem ser criados.
- Exemplo:
  - A classe Emprestimo no diagrama de comunicação do método (operação) *devolverCopia*.


Controlador
Fachada
1. * [achou=false] e:=proximo()
: Emprestimo
devolverCopia(codCopia)--->
: Leitor
2. achou:=devolverCopia(codCopia)
2.1. *[para cada] linha:=proximo()
: LinhaDeEmprestimo
e :
Emprestimo
2.2. cc:=codigoCopia()
2.3. [cc=codCopia] atualizaDataDev(dataDeHoje)
2.2.1. cc = codigoCopia()
linha :
LinhaDeEmprestimo
cop :
CopiaDeLivro
2.3.1. sinalizaDevolucao(dataDeHoje)

Implementação de devolverCopia(codCopia)

```
public class Leitor
{
  private String nome;
  private Char[] tipo;
  private Boolean achou=false;
  private List emprestimos = new ArrayList();
  public void devolverCopia(int codCopia)
   {
    Iterator i = emprestimos.iterator();
    while (i.hasNext()) && (!achou) {
       Emprestimo e = (Emprestimo) i.next();
       achou=e.devolverCopia(codCopia)}
   }
}
```

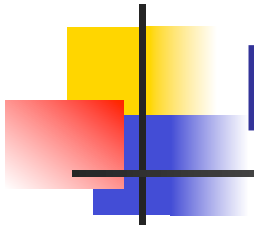
Implementação de devolverCopia(codCopia)

```
public class Emprestimo
{
  private Date data_de_emprestimo;
  private Char[] situacao;
  private int cc=0;
  private List linhas = new ArrayList();
  ...
```

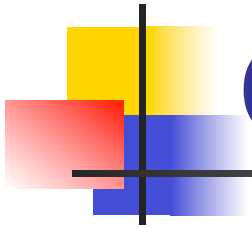
Implementação de devolverCopia(codCopia)

```
....
public Boolean devolverCopia(int codCopia) {
    private Boolean ach = false;
    Iterator i = linhas.iterator();
    Date dataDeHoje = new Date();
    while (i.hasNext()) && (!ach) {
        LinhaDeEmprestimo linha = (LinhaDeEmprestimo) i.next();
        cc=linha.codigoCopia();
        if (cc==codCopia) {
              linha.atualizaDataDev(dataDeHoje);
              ach := true;}
    }
    return ach; }
}
```

Exceções e Tratamento de Erros



- Até aqui...
  - tratamento de erros foi ignorado (intencionalmente)
  - mas deve ser considerado na fase de projeto em sistemas reais
- Exemplo:
  - os contratos podem ser anotados com observações sobre situações típicas de erros e o plano geral de tratamento desses erros.
- A UML não tem uma notação especial para ilustrar exceções.



Ordem de Implementação

- Podem ser implementadas e testadas na seguinte ordem:
  - Das classes com acoplamento mais baixo para as classes com acoplamento mais alto.
- Exemplo:
  - Começar por Livro ou Leitor. Em seguida, as classes que dependem de implementações prévias: Emprestimo ou LinhaDeEmprestimo, e assim por diante.

Do Projeto à Codificação



SSC-526 – Análise e Projeto Orientados a Objeto
Profa. Dra. Elisa Yumi Nakagawa
2º semestre de 2017