

Contratos



Modelagem Funcional



Modelagem Funcional

- Especificação das funções externas do sistema
 - Operações de Sistema - *inputs*
 - Consultas de Sistema - *outputs*



Artefatos necessários

- Modelo conceitual
- Diagramas de seqüência ou casos de uso expandidos



Contrato de Operação de Sistema

- Pré-condições
- Pós-condições
- Exceções



Contrato de Consulta de Sistema

- Pré-condições
- Resultados



Pré-condições

- definem o que deve ser verdadeiro na estrutura da informação armazenada para que a operação ou consulta possa ser executada
- elas não serão testadas durante a execução
- algum mecanismo externo deverá garantir sua validade antes de habilitar a execução da operação ou consulta de sistema correspondente



Pós-condições

- estabelecem o que uma operação de sistema muda na estrutura da informação armazenada



Resultados

- Conjunto de informações retornado por uma consulta




Exceções

- eventos que, se ocorrerem, impedem o prosseguimento correto da operação
- usualmente não podem ser garantidas a priori
- serão testadas *durante* a execução



Tipos de Pré-condições

- *Garantia de parâmetros*: pré-condições que garantem que os parâmetros da operação ou consulta correspondem a elementos válidos do sistema de informação
- *Restrição complementar*: pré-condições que garantem que a informação se encontra em uma determinada situação desejada



Pré-condição de garantia de parâmetros é semântica

- Verificações sintáticas são feitas por tipagem
- Ex.: Ao invés de escrever “x deve ser maior do que zero”, usar `x:InteiroPositivo` na declaração do parâmetro
- Uma pré-condição é semântica se para testa-la for necessário consultar informações gerenciadas pelo sistema



Garantia de Parâmetros

Classe Videolocadora

operação: identificaCliente(nomeCliente:String)

pré:

Existe uma instância da classe *Cliente* tal que o atributo *nome* desta instância é igual ao parâmetro *nomeDoCliente*.

`self.cadastro→exists(nome=nomeCliente)`



Em um contexto não ambíguo

- é possível simplificar a escrita da pré-condição

Classe Videolocadora

operação: identificaCliente(nomeCliente:String)


pré:

Existe um *Cliente* cujo *nome* é igual a *nomeDoCliente*.



Restrição complementar

- exemplo: se o modelo conceitual especifica que uma associação tem multiplicidade de papel 0..1, uma pré-condição complementar poderá especificar que, para uma instância específica, a associação efetivamente existe (ou não existe)
- uma pré-condição nunca poderá contradizer as especificações do modelo conceitual, apenas complementá-las



Tipos de restrições complementares

- Existe (ou não existe) uma instância (ou um conjunto de instâncias) com determinadas propriedades.
- Todas as instâncias (ou nenhuma das instâncias) de uma determinada classe (ou um conjunto definido por uma associação) têm determinadas propriedades.
- Uma associação não obrigatória (com multiplicidade de papel 0..1 ou *) existe (ou não existe) entre determinadas instâncias.
- Um determinado atributo de uma instância tem um certo valor.



Exemplo

Classe Videolocadora

operação: identificaCliente(nomeCliente:String)

pré:

Existe um *Cliente* cujo *nome* é igual a *nomeDoCliente*.

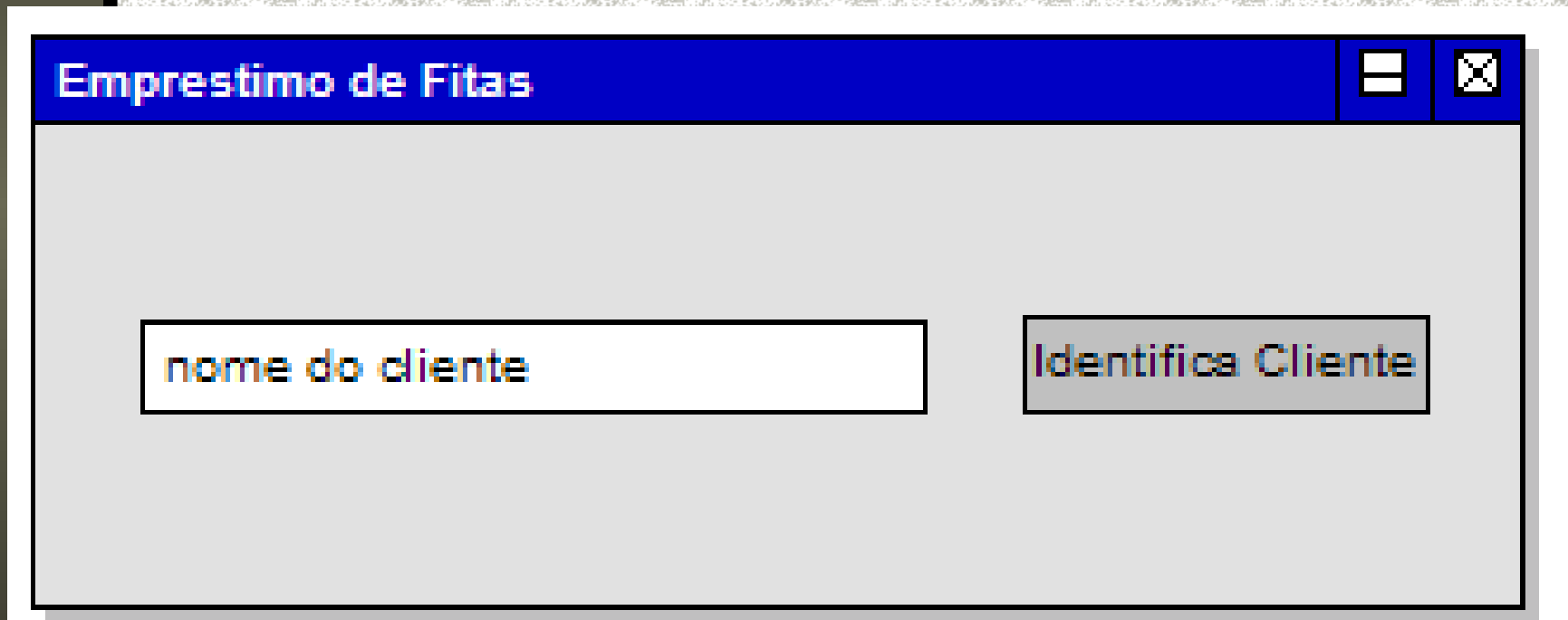
Este *Cliente* possui *débito* igual a zero.

self.cadastro→select(nome=nomeCliente).debito=0



Como garantir pré-condições

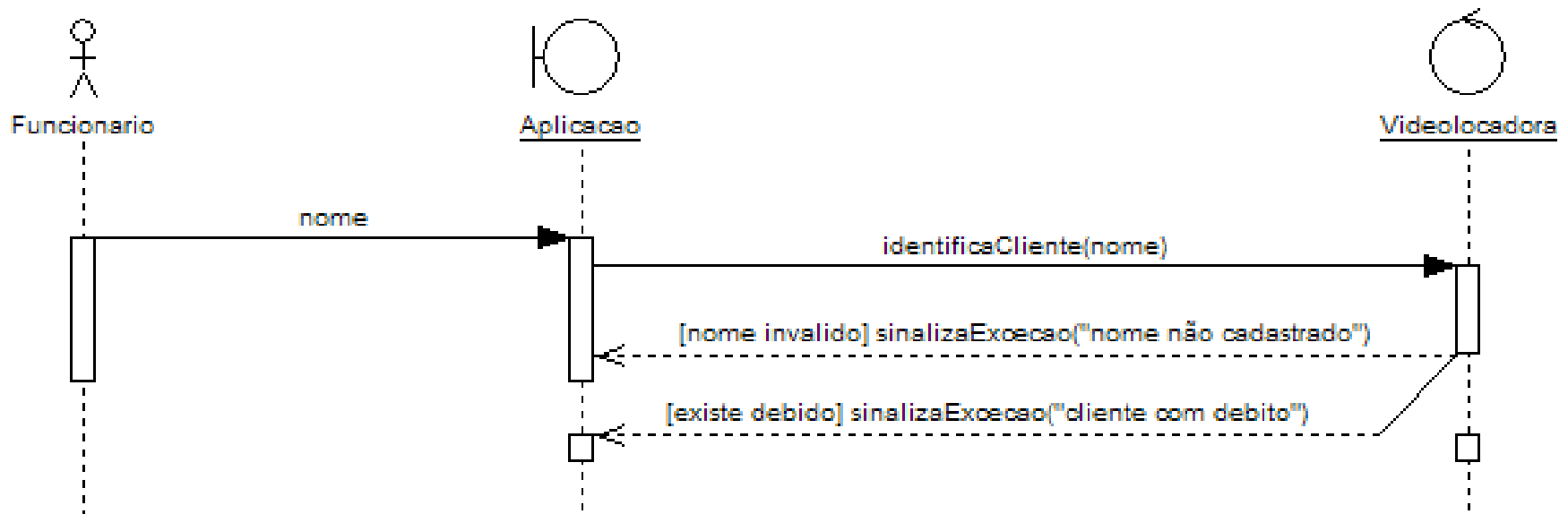
- Considere a seguinte interface



The image shows a screenshot of a software interface window titled "Emprestimo de Fitas". The window has a blue title bar with standard Windows-style control buttons (minimize, maximize, close) on the right. The main area of the window is light gray and contains a text input field with the placeholder text "nome do cliente" and a button labeled "Identifica Cliente".

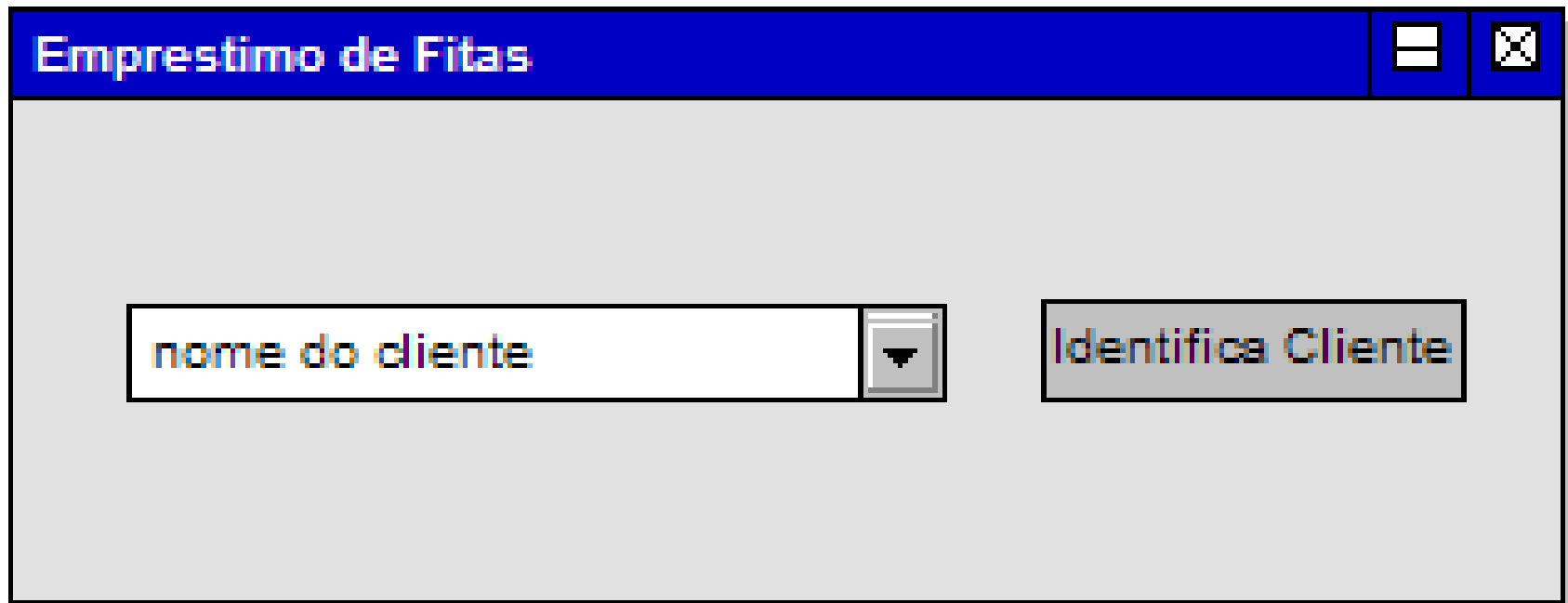
Emprestimo de Fitas	
<input type="text" value="nome do cliente"/>	<input type="button" value="Identifica Cliente"/>

Diagrama de seqüência com exceções



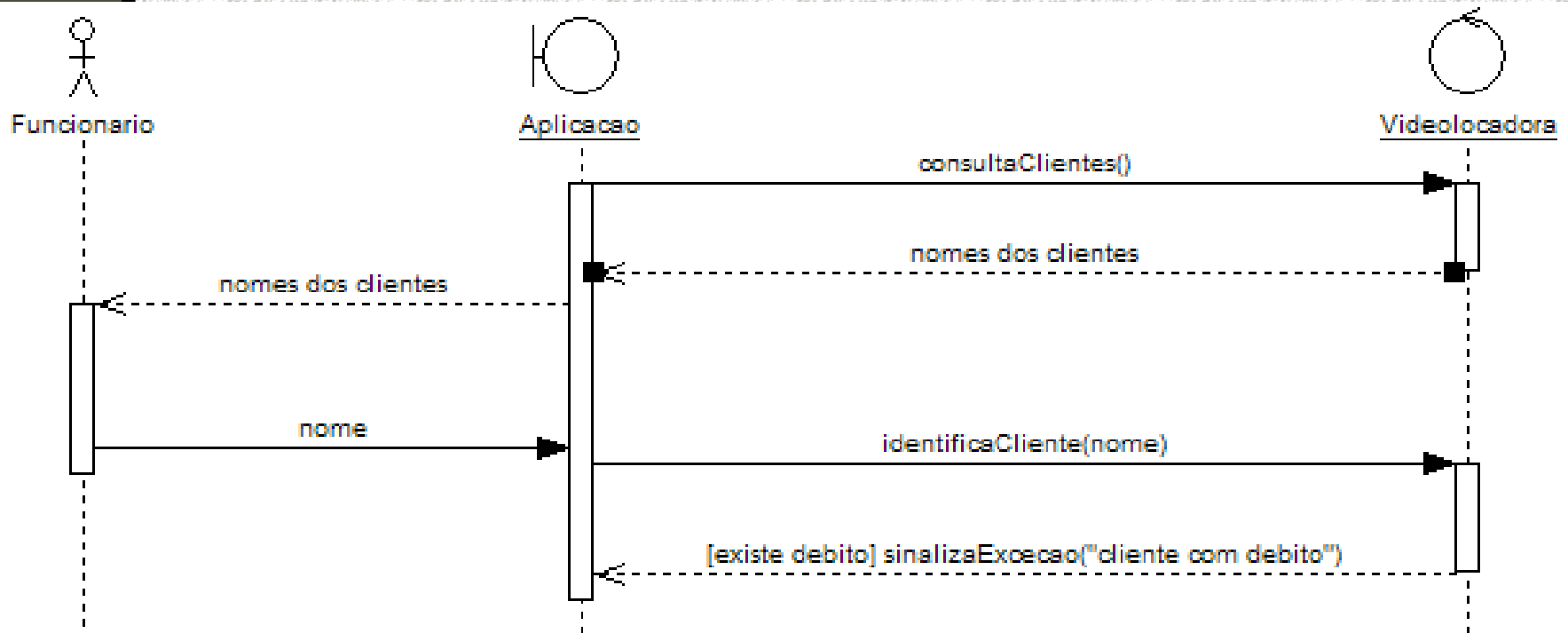
Convertendo uma possível exceção em pré-condição

- Interface modificada

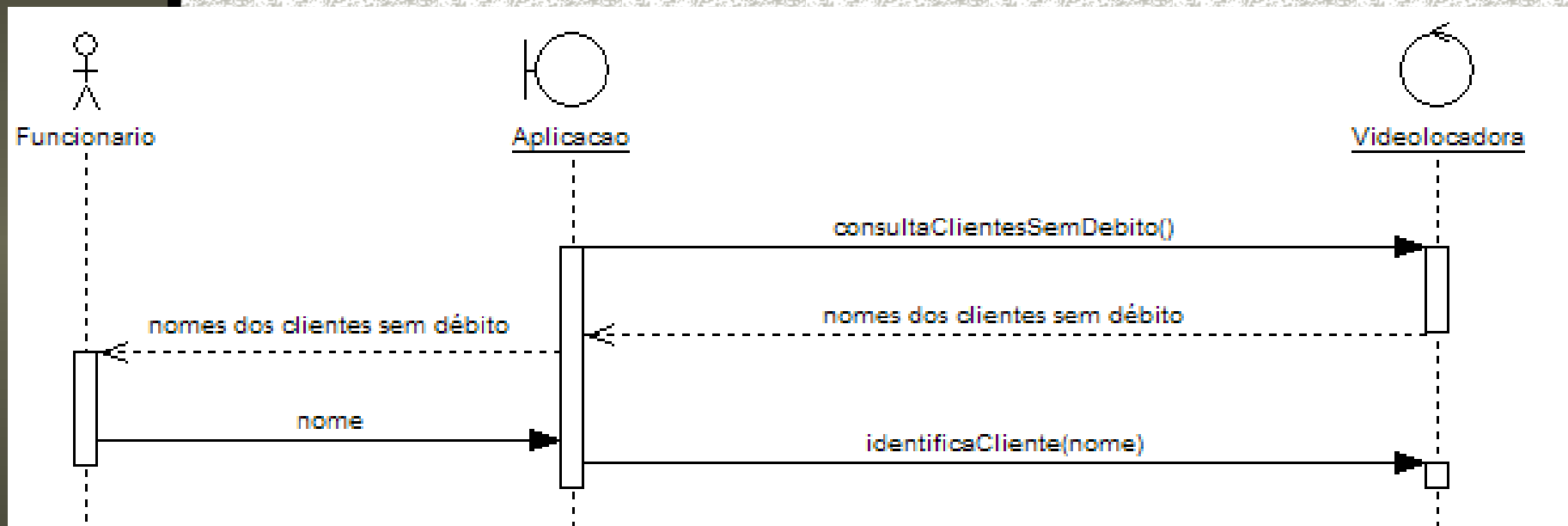


The image shows a screenshot of a Java Swing window titled "Emprestimo de Fitas". The window has a blue title bar with standard window controls (minimize, maximize, close) on the right. The main content area is light gray. It contains a text input field with the placeholder text "nome do cliente" and a small downward arrow button to its right. To the right of the input field is a button labeled "Identifica Cliente".

DS modificado para garantir uma pré-condição

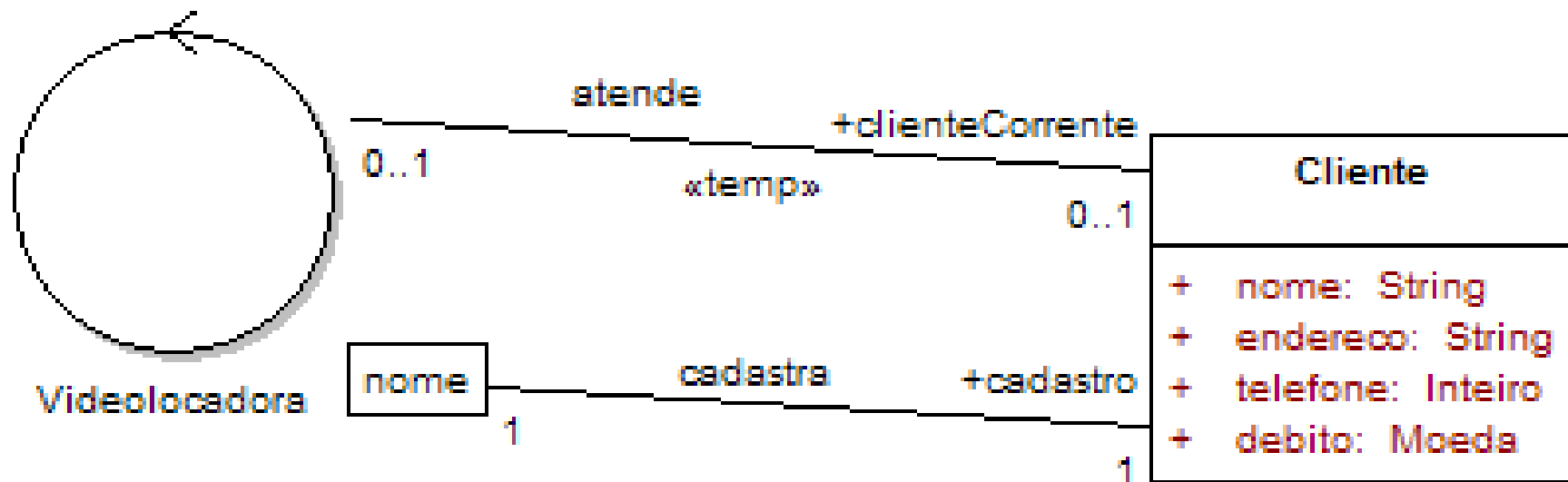



DS modificado para garantir ambas



Associações temporárias

- são usadas para representar informações que não precisam persistir





Uso da associação temporária no contrato

Classe Videolocadora

operação: emprestaFita(codigoFita:String)

pré:


Existe um *clienteCorrente*.

`self.clienteCorrente→size=1`



Pós-condições semânticas

- Instância: criação e destruição
- Associação: criação e destruição
- Atributo: modificação de valor



Criação de uma instância e sua associação com outra instância preexistente

- exemplo: “foi criado um Cliente e associado à Videolocadora por ‘cadastra’”
- ou ainda, fazendo referência ao papel e não à associação, “um novo Cliente foi adicionado em cadastro”.
- em OCL :
self.cadastro→including(Cliente.new).




Destruição de uma instância

- exemplo: “foi destruído um Cliente cujo nome é igual a nomeCliente”.
- Presume-se que quando uma instância é destruída, todas as associações ligadas a ela também o sejam.
- em OCL:
`self.cadastro→excluding(nome=nomeCliente)`
- Deve-se tomar cuidado com questões estruturais (associações obrigatórias) quando um objeto é destruído.




Criação de uma associação entre duas instâncias

- exemplo: “foi criada uma associação ‘atende’ entre a Videolocadora e o Cliente cujo nome é igual a nomeCliente”
- ou, fazendo referência ao papel da associação, “o Cliente cujo nome é igual a nomeCliente foi tornado clienteCorrente”.
- em OCL: `self.clienteCorrente=self.cadastro→select(nome=nomeCliente).`



Destruição de uma associação

- exemplo: “foi destruída a associação atende”
- em OCL: “self.clienteCorrente→size=0”

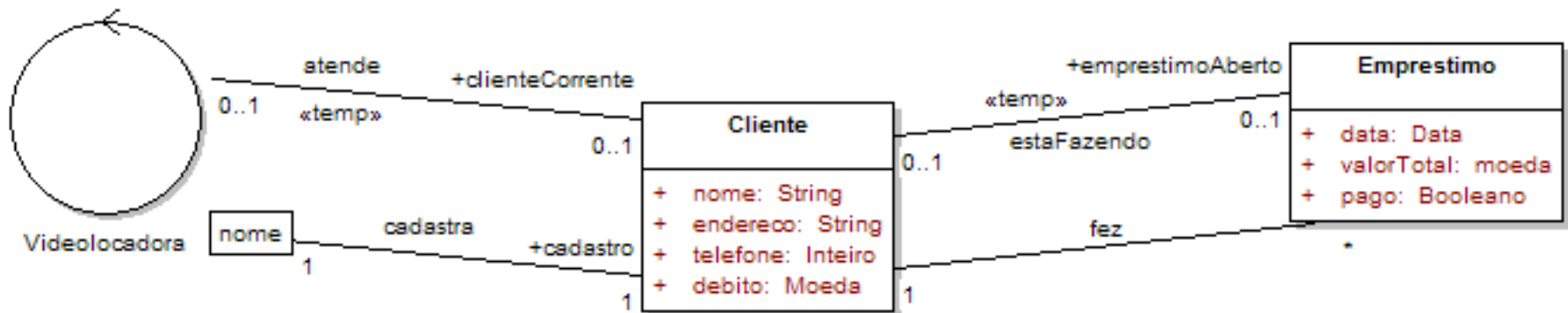


Modificação do valor de um atributo de uma instância

- exemplo: “O debito do clienteCorrente foi alterado para 50,00”
- em OCL:
`self.clienteCorrente.debito=50,00`

Pós-condição condicional

- exemplo: “se não houver nenhum emprestimoAberto associado ao clienteCorrente, então um novo Emprestimo foi criado e associado ao clienteCorrente como emprestimoAberto”



- `self.clienteCorrente.emprestimoAberto@pre→size=0`
`IMPLIES self.clienteCorrente.emprestimoAberto=`
`Emprestimo.new()`



Exceções

Classe Videolocadora

operação: identificaCliente(nomeC:String)

pré:

Existe um *Cliente* com nome igual a nomeC.

pós:

O *Cliente* foi associado como *clienteCorrente*.

exceções:

“Cliente com pendência”: O debito do *Cliente* é diferente de zero.

- `self.cadastro→select(nome=nomeC).debito<>0`



Contrato para Inserção

Classe Videolocadora

operação: cadastraCliente(nomeC,enderecoC,telefoneC:String)

pré:

Não existe nenhum *Cliente* com *nome* = *nomeC*.

pós:

Foi criado um *Cliente* e adicionado ao *cadastro*.

Os atributos *nome*, *endereco* e *telefone* do *Cliente* foram alterados para *nomeC*, *enderecoC* e *telefoneC*.

O atributo *debito* do *Cliente* foi alterado para 0,00.



Em OCL

Classe Videolocadora

operação: cadastraCliente(nomeC,enderecoC,telefoneC:String)

pré:

self.cadastro→select(nome=nomeC)→size=0

pós:

self.cadastro→including(Cliente.new(nomeC,enderecoC,telefoneC, 0))



Contrato para Alteração

Classe Videolocadora

operação: alteraCliente(nomeC,enderecoC,telefoneC:String)

pré:

Existe um *Cliente* com *nome* = *nomeC*.

pós:

Os atributos *endereco* e *telefone* do *Cliente* foram alterados para *enderecoC* e *telefoneC*.



Em OCL

Classe Videolocadora

operação: alteraCliente(nomeC,enderecoC,telefoneC:String)

pré:

self.cadastro→select(nome=nomeC)→size=1

pós:

self.cadastro→select(nome=nomeC).endereco = enderecoC

self.cadastro→select(nome=nomeC).telefone = telefoneC



Contrato para Exclusão

Classe Videolocadora

operação: excluiCliente(nomeC:String)

pré:

Existe um *Cliente* com *nome* = *nomeC*.

Esse *Cliente* não possui associação com nenhum *Emprestimo*.

pós:

O *Cliente* com *nome* = *nomeC* foi destruído.



Em OCL

Classe Videolocadora

operação: excluiCliente(nomeC:String)


pré:

self.cadastro→select(nome=nomeC)→size=1.

self.cadastro→select(nome=nomeC).emprestimos→size=0.


pós:

self.cadastro→select(nome=nomeC)→size=0.




Se a instância a ser destruída tem associação obrigatória com Cliente

- Colocar como pré-condição da operação a não existência de associação do Cliente com Reserva, e incluir um mecanismo de interface que só habilite para exclusão os clientes que não tem reserva (como foi feito em relação aos empréstimos).



Se a instância a ser destruída tem associação obrigatória com Cliente

- Colocar como exceção da operação a verificação da existência de alguma reserva. Neste caso, se o usuário tentar excluir algum cliente que tenha alguma reserva em seu nome a operação não será concluída, e uma exceção será disparada, possivelmente causando a exibição de uma mensagem de erro.



Se a instância a ser destruída tem associação obrigatória com Cliente

- Fazer a ação de destruição do Cliente se propagar para as reservas em seu nome. Neste caso, deveria ser adicionada ao contrato uma pós-condição do tipo: “todas as reservas associadas ao cliente foram excluídas”.



Contrato para Consulta

Classe Videolocadora

consulta: consultaCliente(nomeC:String)

pré:

Existe um *Cliente* com *nome* = *nomeC*.

retorno:

Retorna o *endereço* e o *telefone* do *Cliente*



Em OCL

Classe Videolocadora


consulta: consultaCliente(nomeC:String)

pré:

self.cadastro→select(nome=nomeC)→size=1

retorno:

self.cadastro→select(nome=nomeC)→collect(Sequence{nome,endereco,telefone})



Contrato para uma Consulta de Listagem

Classe Videolocadora

consulta: listaClientes()

pré:

retorno:

retorna os *nomes* de todos os *Clientes* do *cadastro*.



Em OCL

Classe Videolocadora

consulta: listaClientes()

pré:

retorno:

self.cadastro→collect(nome)

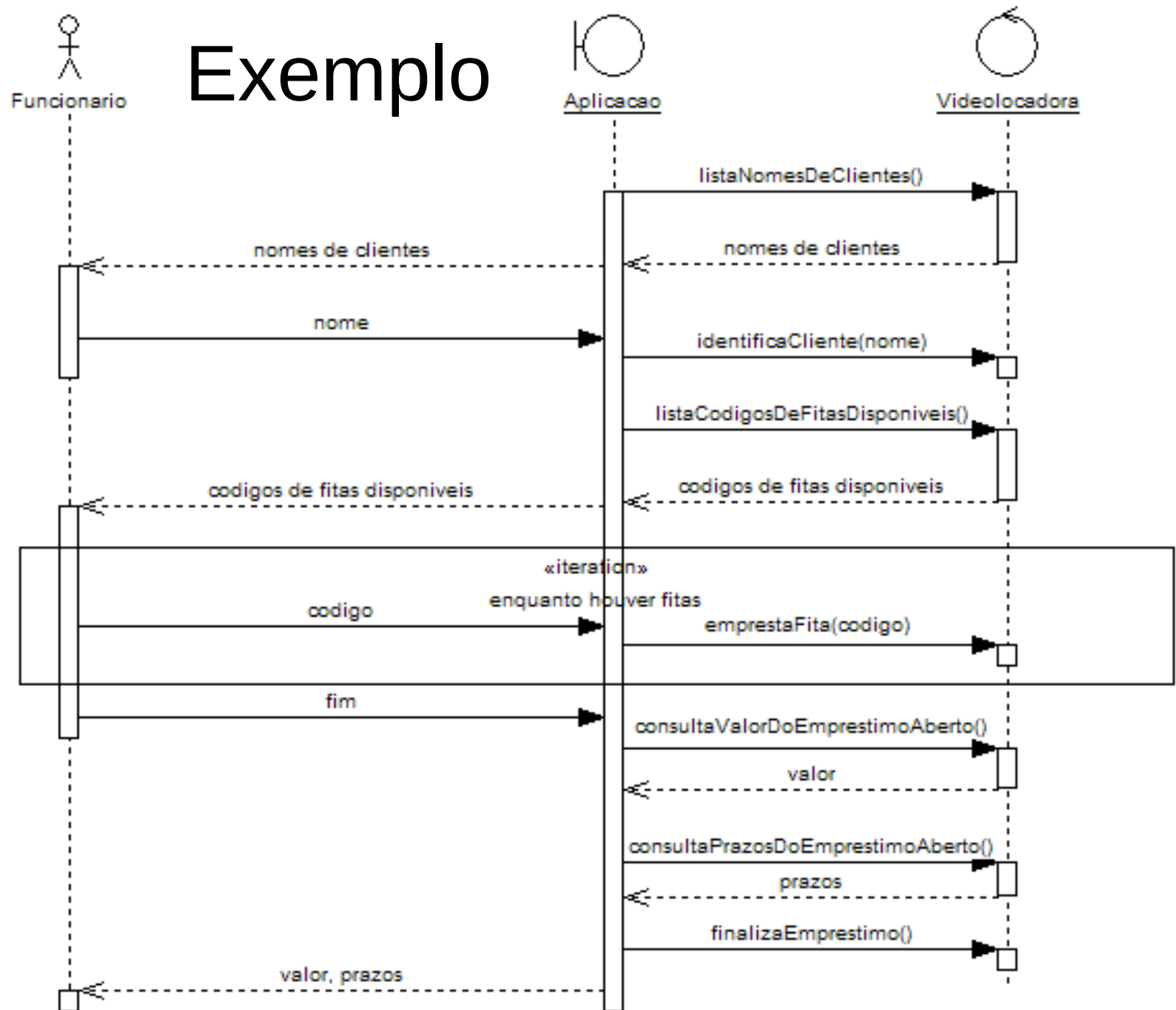


Consultas e Operações Relacionadas com Casos de Uso

- freqüentemente haverá uma cadeia de execução ao longo de um dos fluxos
- Verificar:
 - Qual é o objetivo de cada operação?
 - O que cada uma delas espera que tenha sido produzido pelas anteriores?
 - O que cada uma delas produz?
 - Que exceções poderiam ocorrer durante a execução?



Exemplo





Classe Videolocadora

consulta: listaNomesDeClientes()

pré:

retorno:

Para cada *Cliente* em *cadastro*:

a) *nome*



Classe Videolocadora

operação: identificaCliente(nomeC:String)

pré:

Existe um *Cliente* com *nome* = *nomeC*

Não existe *clienteCorrente*

pós:

O Cliente com *nome* = *nomeC* foi associado à Videolocadora como *clienteCorrente*



Classe Videolocadora

consulta: listaCodigosDeFitasDisponiveis()

pré:

retorno:

Para cada *Fita* que não esteja associada por *apareceEm*:

a) *codigo*

Classe Videolocadora

operação: emprestaFita(codigoF:String)

pré:

Existe um *clienteCorrente*

Existe uma *Fita* com *codigo* = *codigoF*

pós:

Se não existia um *emprestimoAberto* para o *clienteCorrente*, então foi criado um *Emprestimo* com *data* igual a data de hoje, *valorTotal* igual a 0,00, e associado como *emprestimoAberto* ao *clienteCorrente*.

Foi criado um *ItemDeEmprestimo* e associado ao *emprestimoAberto*.

Foi criado um *EstadoDeItemDeEmprestimo EmAndamento* e associado ao *ItemDeEmprestimo*.

O *EstadoDeItemDeEmprestimo EmAndamento* foi associado à *Fita*.

O *prazo* do *ItemDeEmprestimo* foi alterado para o *prazo* do *TipoDeFilme* do *Filme* da *Fita*.

O *valor* do *ItemDeEmprestimo* foi alterado para o *valor* do *TipoDeFilme* do *Filme* da *Fita*.

O *valorTotal* do *emprestimoAberto* foi alterado para o *valorTotal* anterior somado ao *valor* do *ItemDeEmprestimo*.



Classe Videolocadora

consulta: consultaValorDoEmprestimoAberto()

pré:

Existe um `emprestimoAberto` para um *clienteCorrente*

retorno:

O `valorTotal` do `emprestimoAberto` do `clienteCorrente`.



Classe Videolocadora

consulta: consultaPrazosDoEmprestimoAberto()

pré:

Existe um `emprestimoAberto` para um *clienteCorrente*

retorno:

Para cada `ItemDeEmprestimo` do `emprestimoAberto`:

- a) O título do Filme da Fita do `EstadoDoItemDeEmprestimo`;
- b) O prazo.



Classe Videolocadora

operação: finalizaEmprestimo()

pré:

Existe um *emprestimoAberto* para um *clienteCorrente*.

O *emprestimoAberto* possui pelo menos um *ItemDeEmprestimo*.

pós:

Foi destruída a associação com *emprestimoAberto*.

Foi destruída a associação com *clienteCorrente*.