Processo de Desenvolvimento de Software

Da Concepção a Elaboração (Iteração 01)

Prof. Carlos Eduardo de B. Paes
Departamento de Computação
Pontifícia Universidade Católica de São Paulo
carlosp@pucsp.br

Requisitos da Iteração 1

- Implementar um cenário-chave básico do caso de uso
  - Processar Venda: entrar itens e receber o pagamento em dinheiro
- Sem colaboração com dispositivos externos (tais como o calculador de impostos ou o banco de dados de produtos)
- Nenhuma regra complexa de preços é aplicada
- Iterações subsequentes serão desenvolvidas sobre essa fundação

Desenvolvimento incremental para o mesmo caso de uso através das iterações

- Nem todos os requisitos no caso de uso Processar Venda estão sendo tratados na iteração 1
- É comum trabalhar em cenários variados ou características do mesmo caso de uso sobre vários cenários e gradualmente estender o sistema para finalmente tratar toda a funcionalidade exigida
- Por outro lado, casos de uso curtos e simples devem ser completados dentro de uma interação

Modelo de Casos de Uso: desenhando Diagramas de Sequencia do Sistema

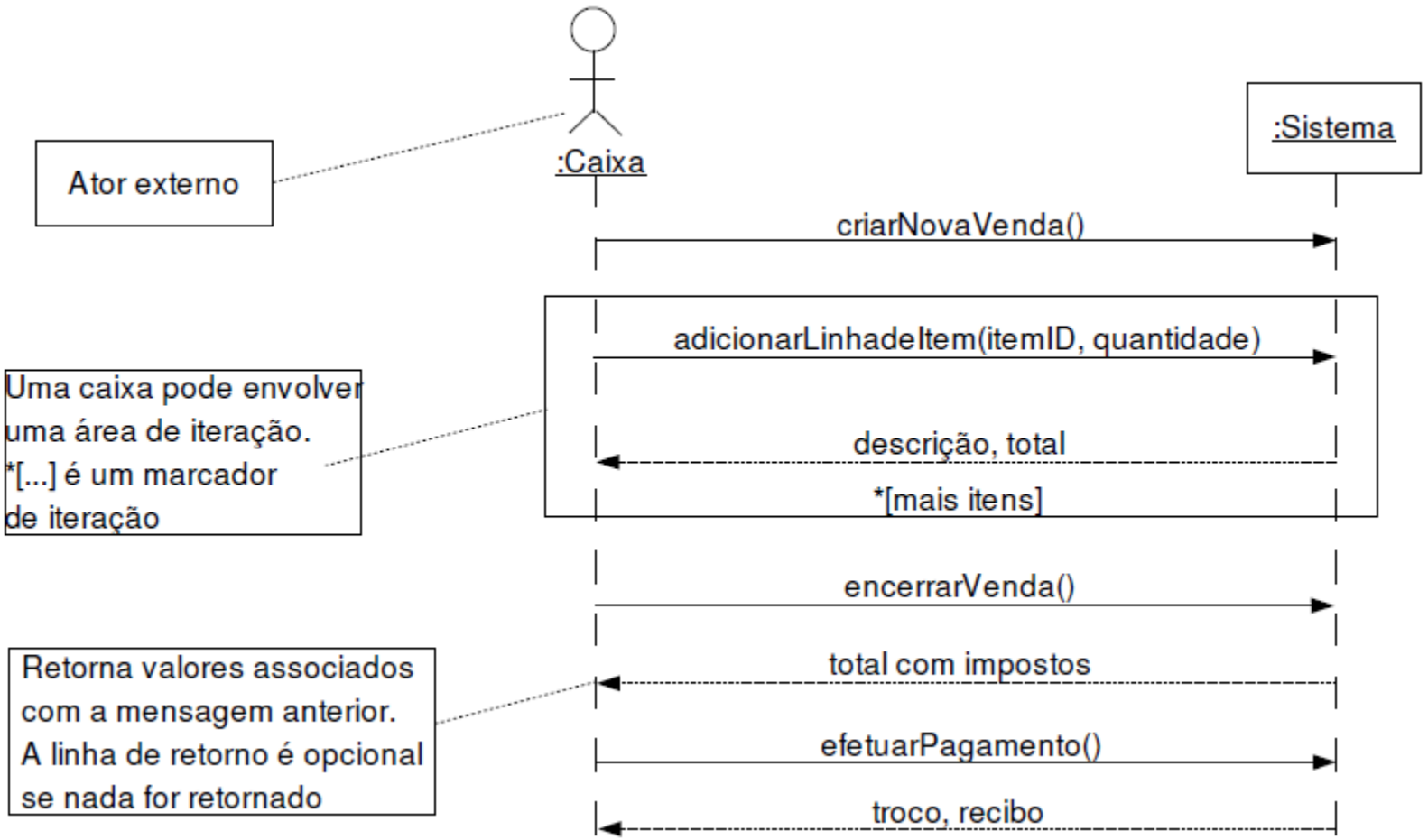
O comportamento do sistema e Diagramas de Sequencia da UML

- É útil investigar e definir o comportamento do software como uma "caixa preta"
- O comportamento do sistema é uma descrição do que o sistema faz (sem uma explicação de como ele o faz)
- Casos de uso descrevem como atores externos interagem com o sistema de software. Durante esta interação, um ator gera eventos
- Um evento de solicitação inicia uma operação sobre o sistema

O comportamento do sistema e Diagramas de Sequencia do Sistema (DSS)

- Um diagrama de sequencia é uma figura que mostra, de um cenário específico de um caso de uso, os eventos que atores externos geram, sua ordem e possíveis eventos intra-sistema
- Todos os outros sistemas são tratados como uma caixa preta; o diagrama enfatiza os ventos que cruzam a fronteira do sistema dos atores para os sistemas

SSD para o cenário de “Processar Venda”

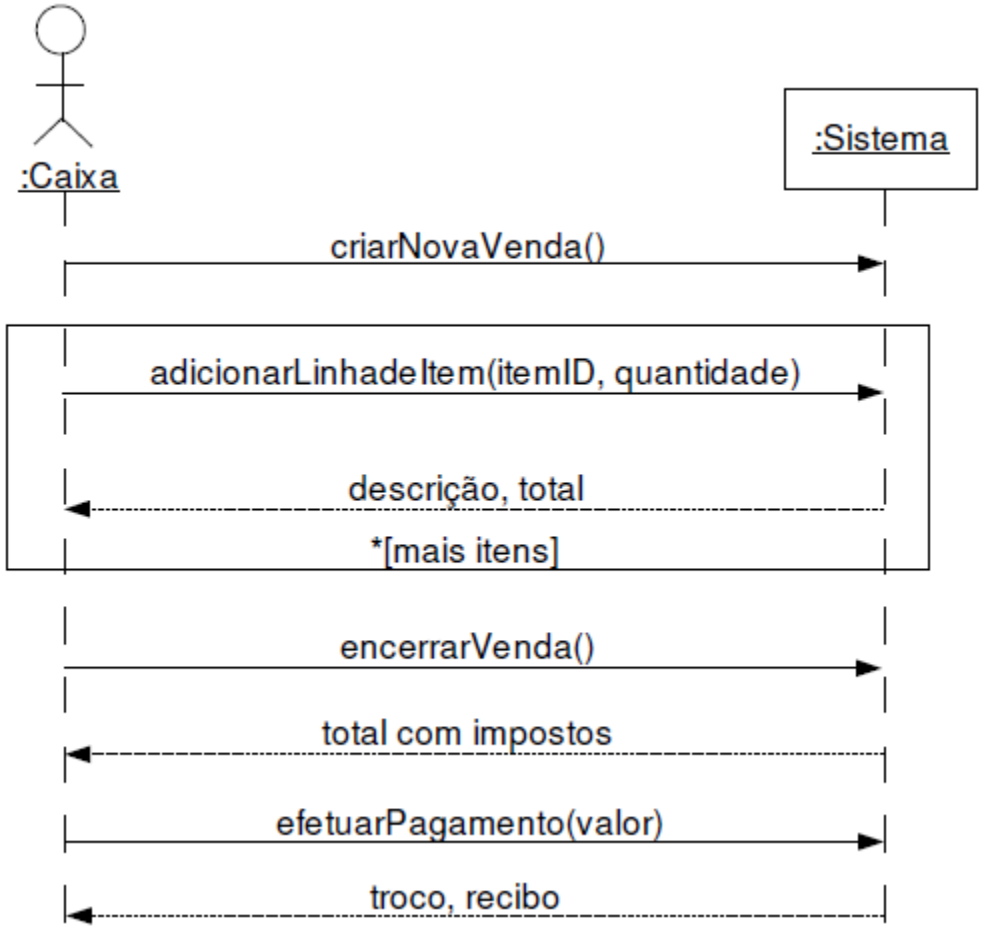


SSD e Casos de Uso

Cenário simples de
Processar Venda - somente dinheiro

1. O cliente chega ao caixa com bens para comprar.
2. O caixa inicia uma nova venda.
3. O caixa insere o identificador do item.
4. O sistema registra a linha de item de venda, e apresenta a descrição do item, o preço e o sub-total.
   O caixa repete os passos 3 e 4 até indicar que está pronto.
4. O sistema apresenta o total com impostos calculados.

...



Nomeando eventos e operações do sistema

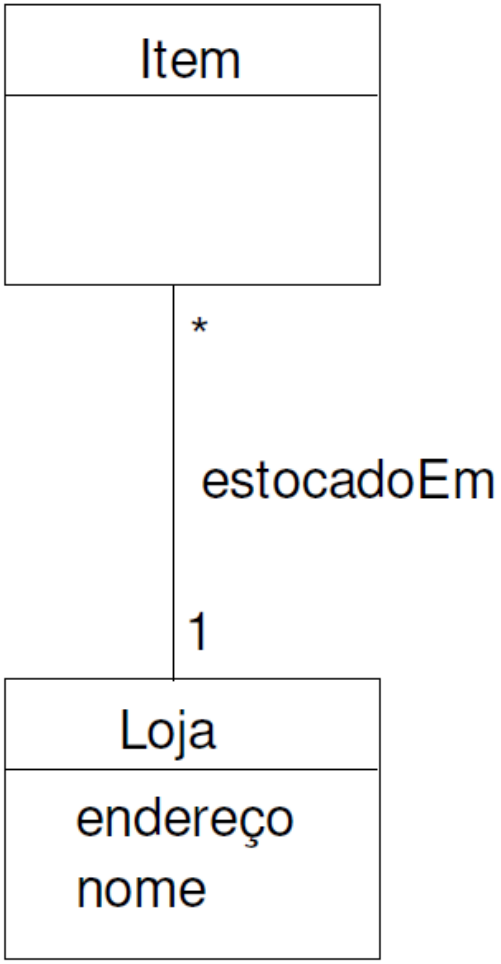
- O conjunto de todas as operações exigidas do sistema é determinado pela identificação dos eventos do sistema
  - criarNovaVenda()
  - adicionarLinhaDeItem(itemID, quantidade)
  - encerrarVenda()
  - efetuarPagamento(valor)

Modelo de Domínio: visualizando conceitos

Modelos de Domínio

- Um Modelo de Domínio ilustra conceitos significativos em um domínio de problema
- Ele é a representação de coisas do mundo real, não de componentes de software
- É um conjunto de diagramas de estrutura estáticos; nenhuma operação é definida
- Ele pode mostrar:
  - Conceitos
  - Associações entre conceitos
  - Atributos dos conceitos

Modelos de Domínio



- Um Modelo de Domínio é uma descrição de coisas do mundo real
- Um Modelo de Domínio não é uma descrição do projeto de software
- Um conceito é uma ideia,coisa ou objeto

Classes Conceituais no domínio de vendas

Modelo de Domínio Parcial

- Uma diferença fundamental entre análise orientada a objetos e análise estrutura: divisão por conceitos (objetos) em vez de divisão por funções

Estratégias para identificar classes conceituais

- Usar uma lista de categorias de classes conceituais
  - fazer uma lista de conceitos candidatos
- Usar identificação de expressões nominais identificar substantivos (ou expressões nominais) em descrições textuais do domínio do problema, e considerá-los como conceitos ou atributos
- **Casos de Uso são descrições excelentes para fazer essa análise**

Usar uma lista de categorias de classes conceituais

Categoria de Conceitos	Exemplo
Objetos físicos ou tangíveis	PDV
Especificações, projetos ou descrições de coisas	EspecificaçãoDeProduto
Lugares	Loja
Transações	Venda, Pagamento
Linhas de itens de transação	LinhaDeItemDeVenda
Papéis de pessoas	Caixa
Contêineres de outras coisas	Armazém, Cesta

- Lista completa: Larman, C, Utilizando UML e Padrões, 3ed

Encontrando classes conceituais com identificação de expressões nominais

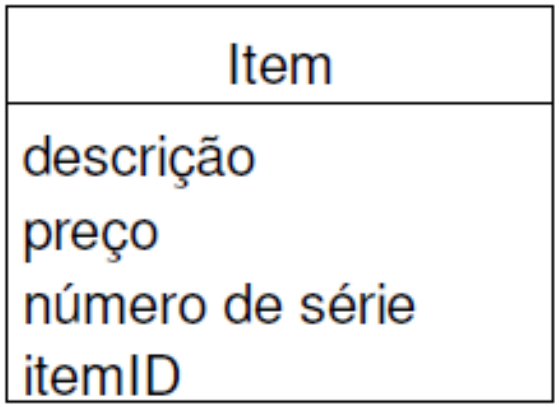
1. Este caso de uso inicia quando um **Cliente** chega em um **PDV** com itens para comprar.
2. O **Caixa** inicia uma nova venda.
3. O **Caixa** insere o **identificador do item**.
4. . ...

• Os Casos de Uso completos são uma descrição excelente para fazer essa análise

• Algumas dessas expressões nominais são conceitos candidatos; outros podem ser atributos dos conceitos

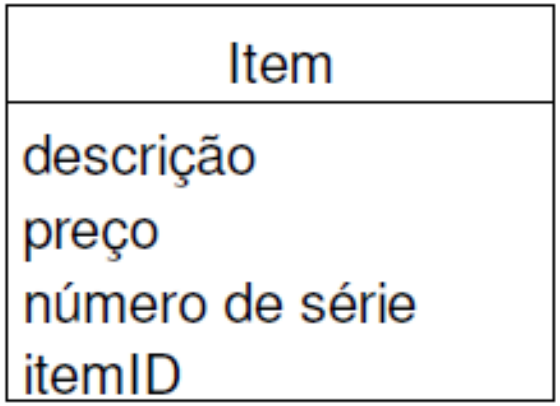
• Um mapeamento substantivo-para-conceito não é possível, porque as palavras na linguagem natural são (algumas vezes) ambíguas

A necessidade de especificar ou descrever classes conceituais



- O que está errado nesta figura?
- Considere a situação em que todos os itens são vendidos, e então excluídos da memória do computador
- Quanto custa um item?

A necessidade de especificar ou descrever classes conceituais



- A memória do preço de um item foi anexada a itens do estoque, que serão deletados
- Perceba também que neste modelo existem dados duplicados (descrição, preço, itemID)

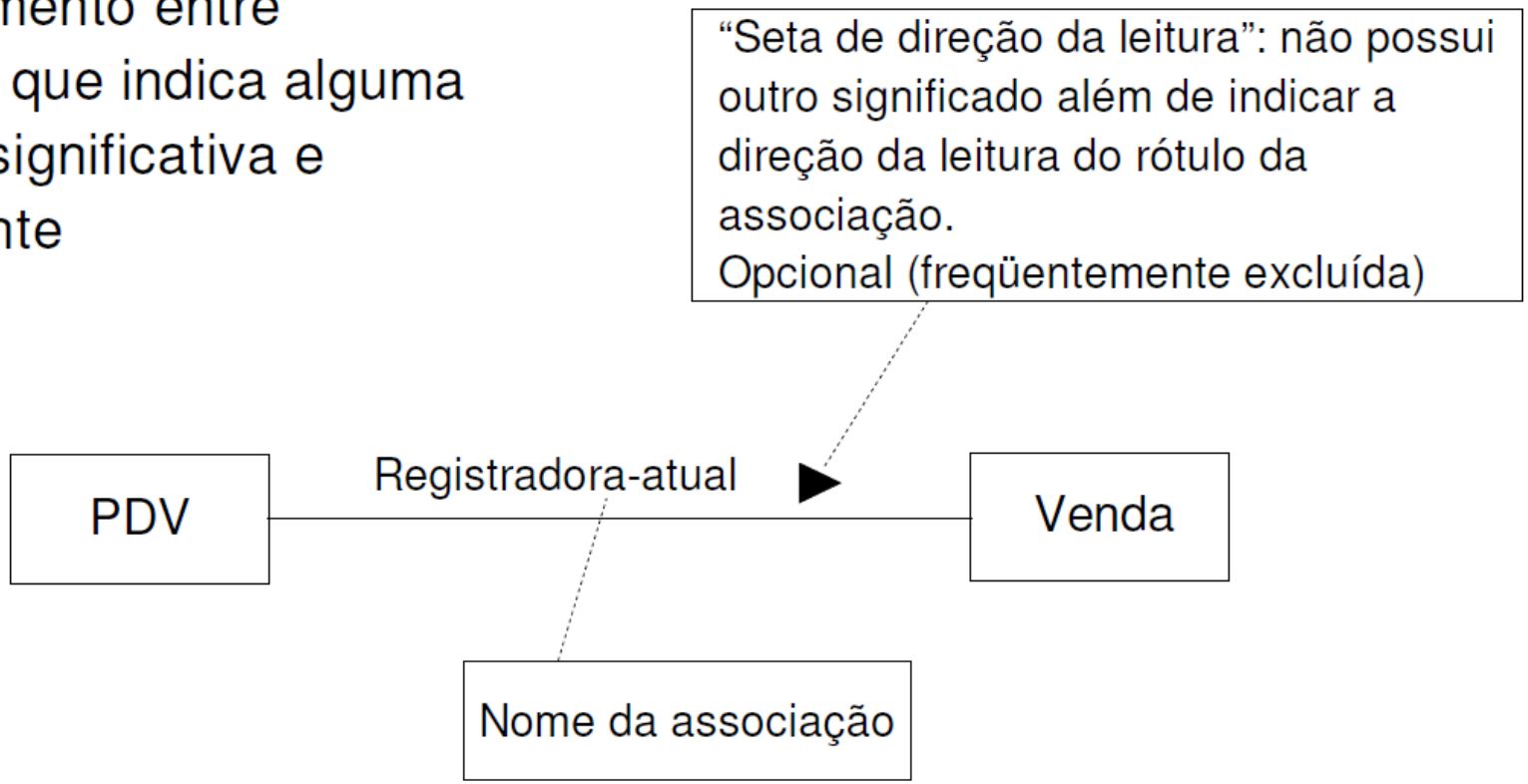
A necessidade de especificar ou descrever classes conceituais

- Adicionar um conceito de descrição ou especificação quando:
  - a exclusão de instâncias das coisas que elas descrevem resultar em perda da informação que necessita ser mantida, devido a associação incorreta de informações com a coisa deletada
  - ele reduz a duplicação ou redundância de informações
- Similar ao processo de normalização em projeto de banco de dados!

O Modelo de Domínio (parcial) do POS NextGen

Adicionando associações

Uma associação é um relacionamento entre conceitos que indica alguma conexão significativa e interessante

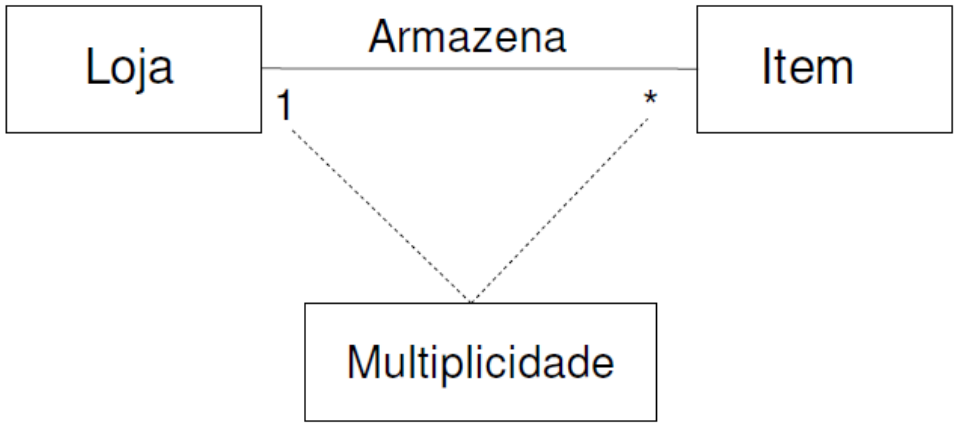


Encontrando associações: lista de associações comuns

Categoria	Exemplos
A é uma parte física de B*	Gaveta – PDV
A é uma parte lógica de B	LinhaDeItemDeVenda – Venda
A está fisicamente contida em B	PDV – Loja
A está logicamente contida em B	DescriçãoDeItem – Catálogo
A é uma descrição de B	DescriçãoDeItem – Item
A é uma linha de item de uma transação ou relatório B	LinhaDeItemDeVenda – Item
A é conhecido/logado/gravado/capturado em B	Venda – PDV
A é um membro de B	Caixa – Loja
...	

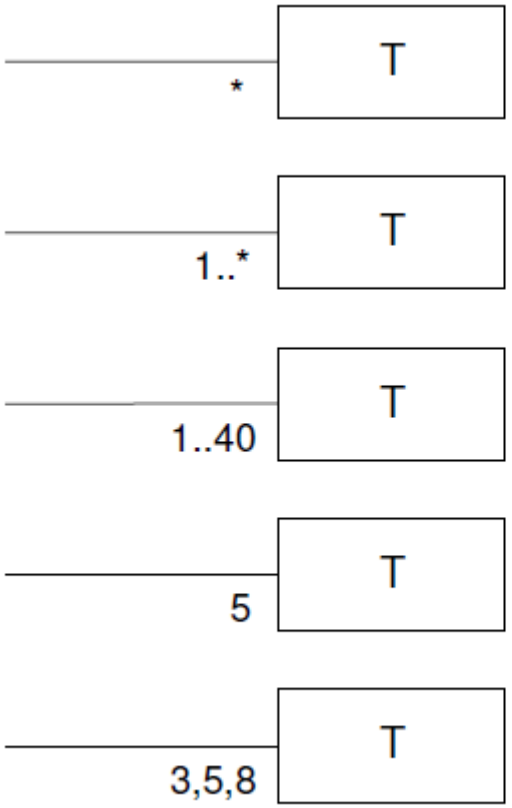
- Lista completa: Larman, C, Utilizando UML e Padrões, 3ed

Multiplicidade



- A multiplicidade define quantas instâncias de um tipo A podem ser associadas a uma instância de um tipo B, em um determinado momento
- Por exemplo, uma única instância de Loja pode ser associada com “muitas” (zero ou mais) instâncias de item

Multiplicidade



Zero ou mais; "muitos"

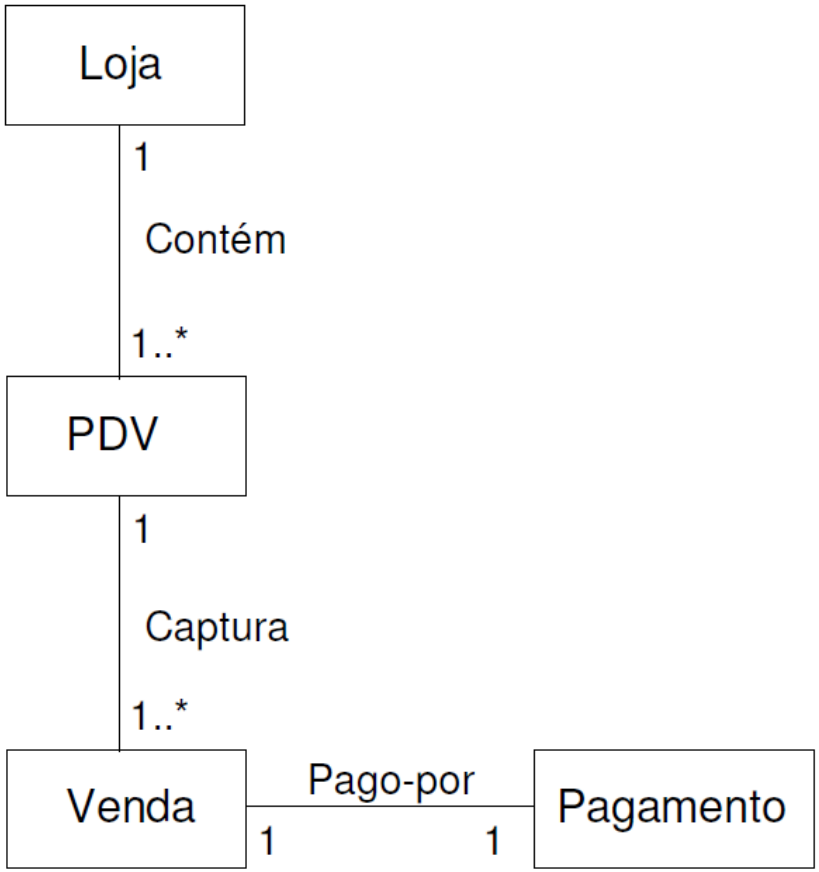
Um ou mais

De um a quarenta

Exatamente cinco

Exatamente três, cinco ou oito

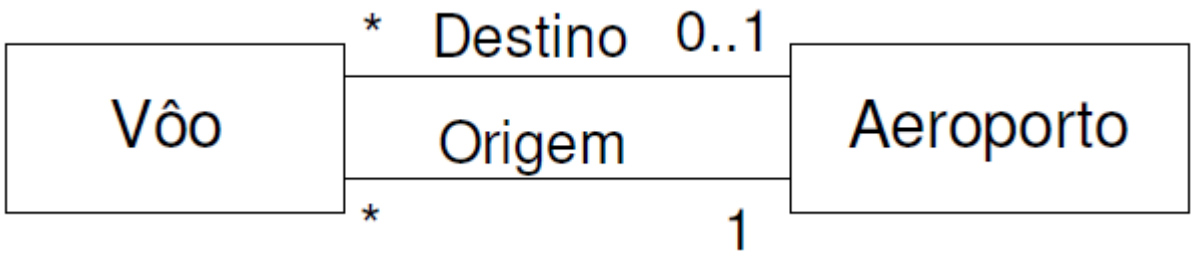
Nomeando associações



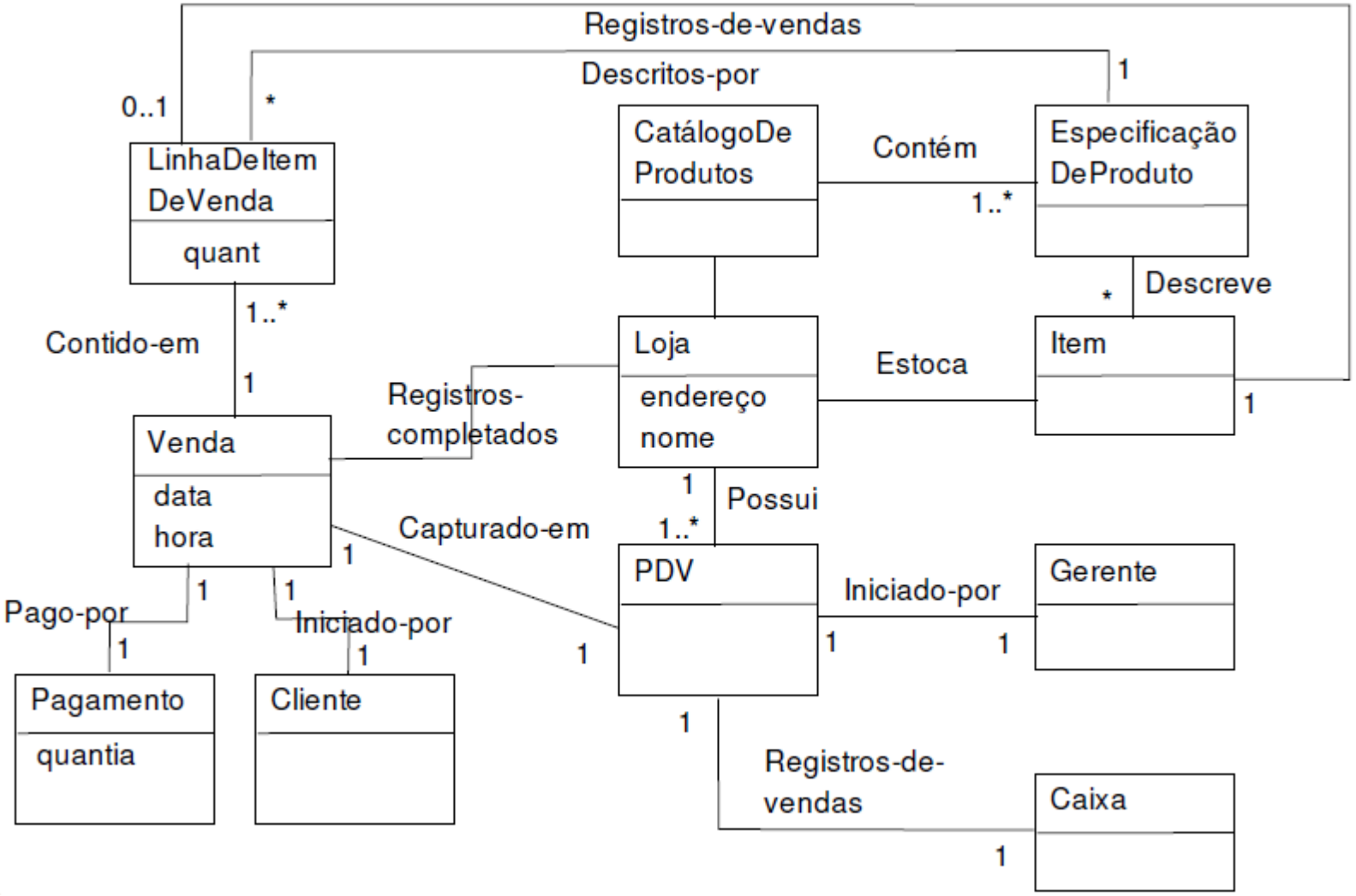
- Nomeie uma associação com base no formato NomeTipo-LocVerb-NomeTipo
- Os nomes das associações devem iniciar com letra maiúscula
- Uma locução verbal deve ser construída com hífens
- A direção default para ler uma associação é da esquerda para a direita, ou de cima para baixo

Múltiplas associações entre dois tipos

- É comum haver múltiplas associações entre dois tipos
- No exemplo, não é garantido que todo voo irá pousar em um aeroporto



Conclusão do Modelo de Domínio



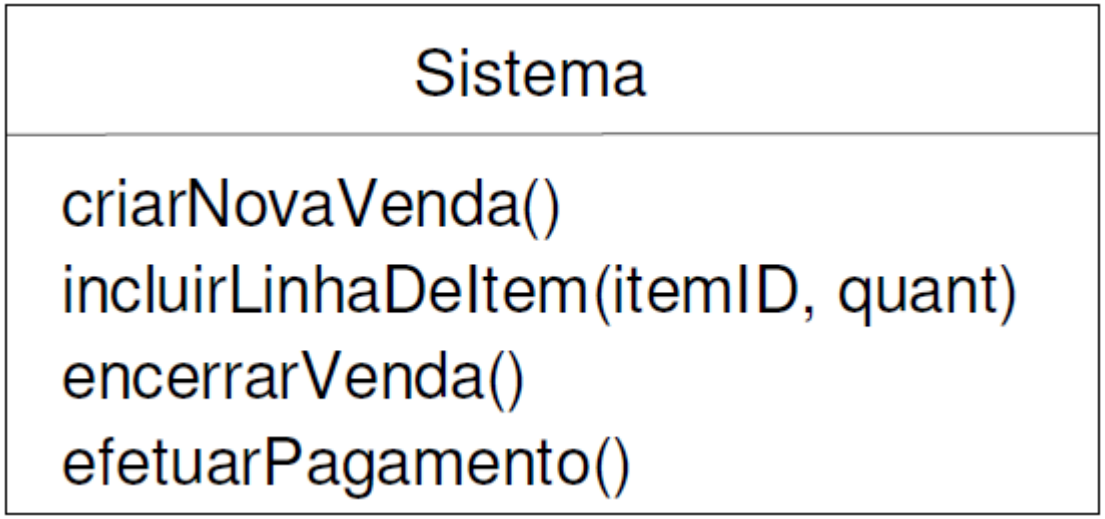
Modelo de Casos de Uso: adicionando detalhes com contratos de operações

Contratos

- Contratos são documentos que descrevem o comportamento do sistema
- Os Contratos podem ser definidos pelas **operações do sistema**
  - operações que o sistema (como uma caixa preta) oferecem em sua interface pública para manipular os eventos do sistema que chegam
- O conjunto completo de operações do sistema distribuído por todos os casos de uso definem a interface pública do sistema

Contratos do Sistema

- Na UML, o sistema como um todo pode ser representado como uma classe
- Os contratos são escritos para cada operação do sistema descrever o seu comportamento



Exemplo de Contrato: incluirLinhaDeItem

Contrato CO2: incluirLinhaDeItem

Operação:	incluirLinhaDeItem(itemID: ItemID, quant: integer)
Referências cruzadas:	Casos de Uso: Processar Venda.
Pré-condições:	Existe uma venda em andamento.

Pós-condições:

- Uma instância de LinhaDeItemDeVenda *sli* foi criada (criação de instância)
- *sli* foi associada com uma venda (associação formada)
- *sli.quant* recebeu o valor de quant (modificação de atributo)
- *sli* foi associada com uma EspecificacaoDeProduto, baseado na combinação do itemID (associação formada)

Pré e Pós-condições

- Pré-condições são suposições a respeito do estado do sistema antes da execução da operação
- Uma pós-condição é uma suposição que se refere ao estado do sistema após a conclusão da operação
  - As pós-condições não são ações a serem executadas durante a operação
  - Descrevem mudanças no estado dos objetos no Modelo de Domínio (instâncias são criadas, associações são formadas ou rompidas e atributos são alterados)

Pós-Condições de incluirLinhaDeItem

- Criação e Remoção de instâncias
- Após o itemID e quantidade de um item serem inseridos pelo caixa, quais novos objetos devem ter sido criados?
  - Uma instância de LinhaDeItemDeVenda **sli** foi criada

Pós-Condições de incluirLinhaDeItem

- Modificação de Atributo
- Após o itemID e a quantidade de um item terem sido inseridos pelo caixa, quais atributos de objetos novos ou existentes devem ter sido modificados?
  - sli.quant recebeu o valor de quant

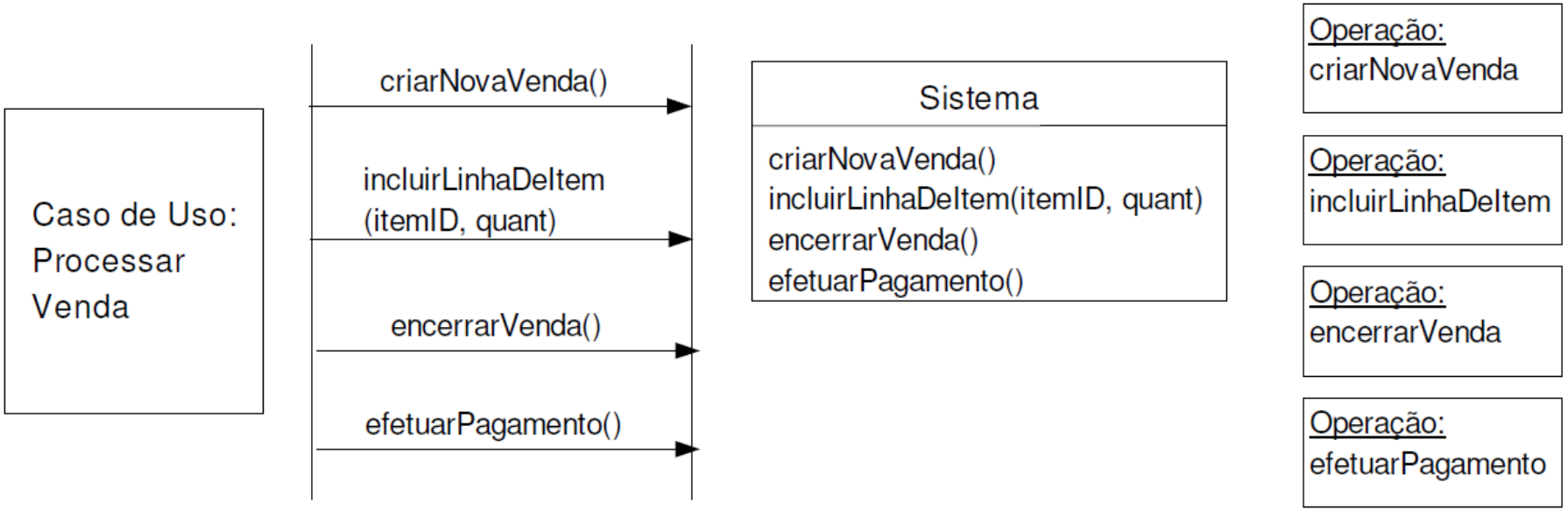
Pós-Condições de incluirLinhaDeItem

- Associações Formadas ou Rompidas
- Após o itemID e a quantidade de um item terem sido inseridos pelo caixa, quais associações entre objetos novos ou existentes devem ter sido formadas ou rompidas?
  - sli foi associada com a venda atual (associação formada)
  - sli foi associada com uma EspecificacaoDeProduto, com base na combinação de itemID (associação formada)

A elaboração de Contratos leva a atualizações no Modelo de Domínio

- Também é comum a descoberta da necessidade de registrar novos conceitos, atributos ou associações no Modelo de Domínio

Linhas-guia para Contratos



Caso de Uso

Diagrama de Seqüência do Sistema

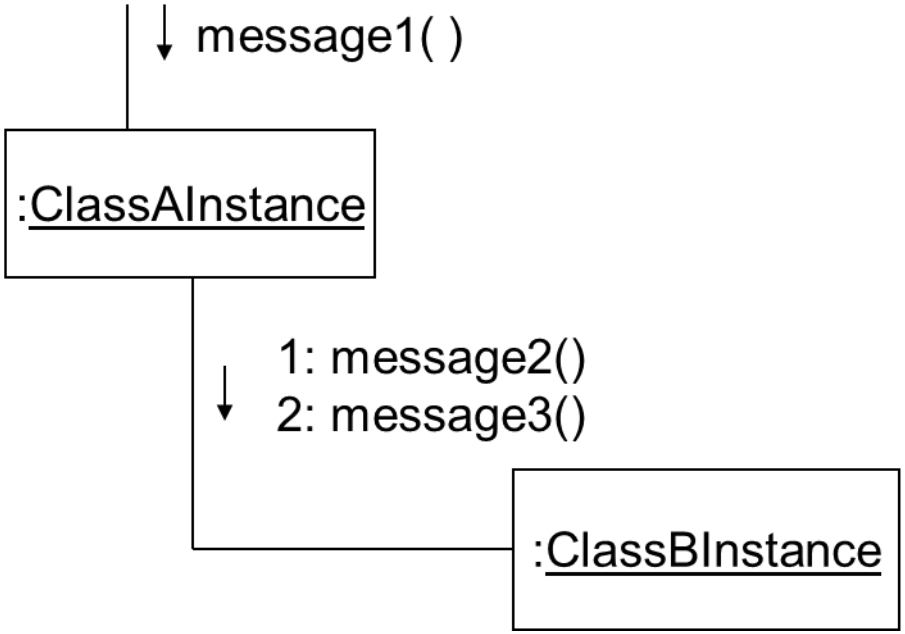
Operações do Sistema

Contratos

Notação dos Diagramas de Interação

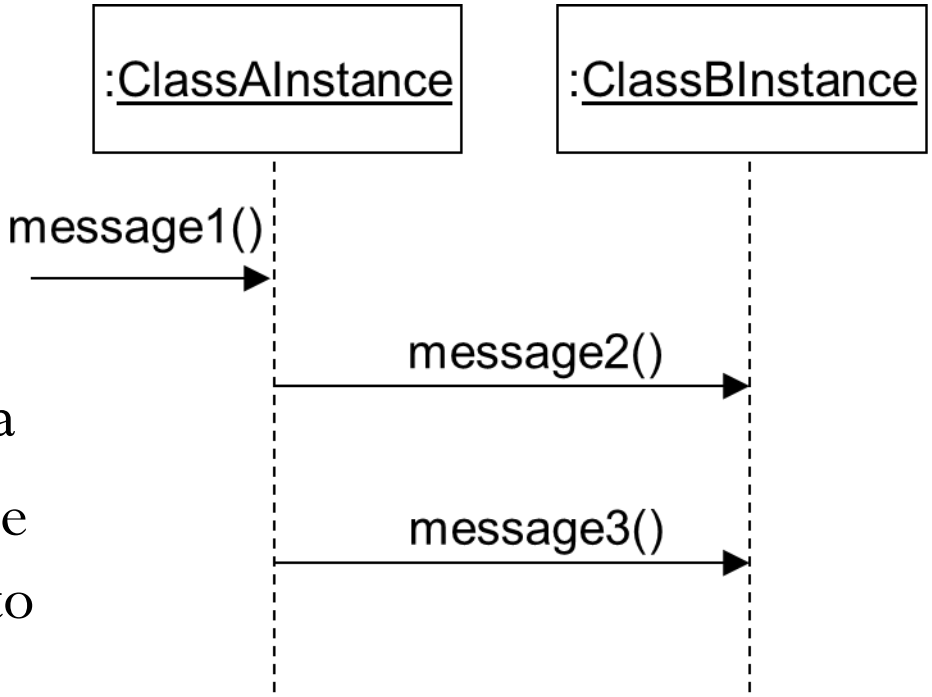
Introdução

- **Diagramas de Interação** ilustram como os objetos interagem por meio de mensagens
- **Diagramas de Colaboração** ilustram as interações entre objetos em forma de grafo ou rede

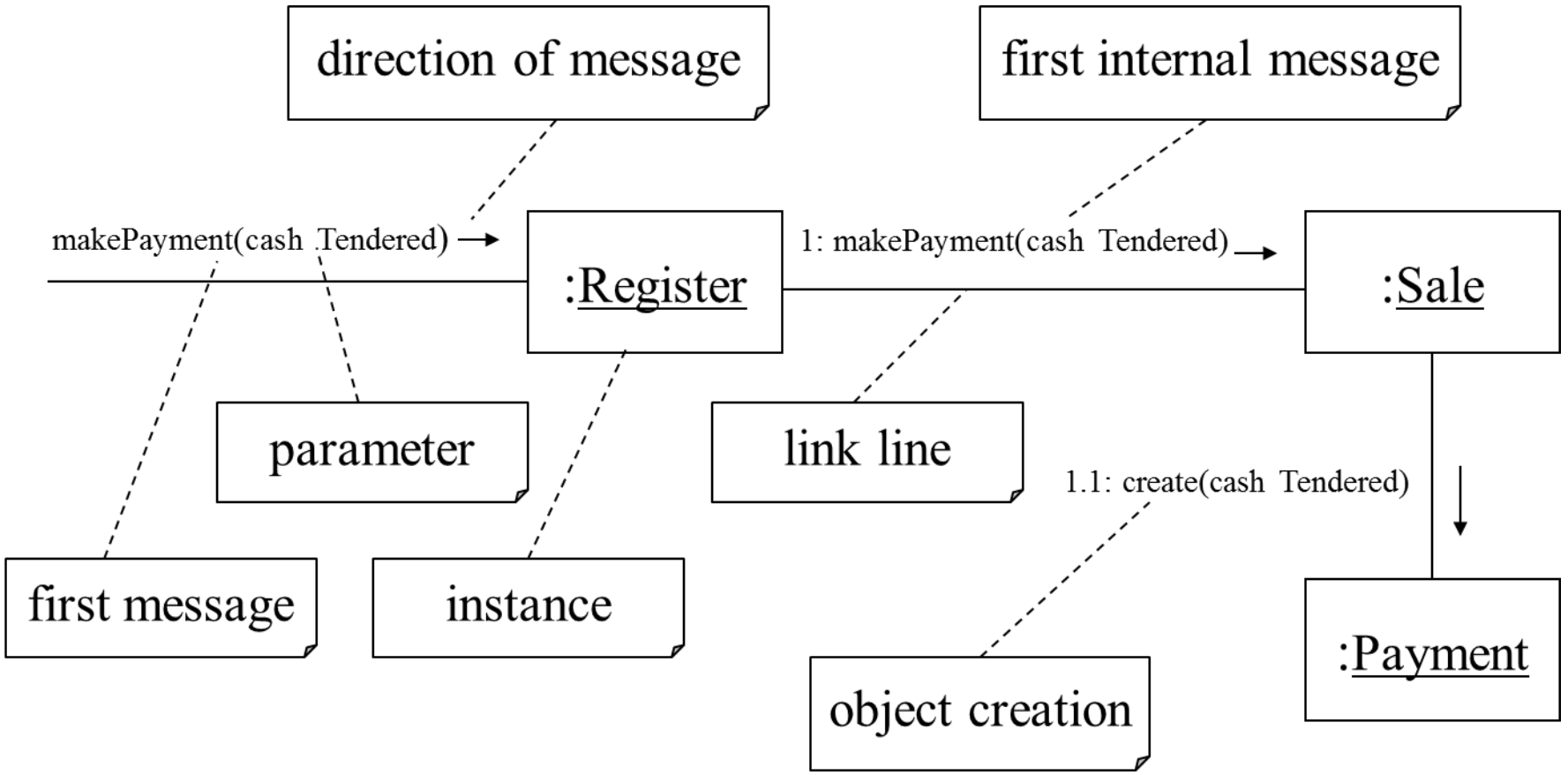


Introdução

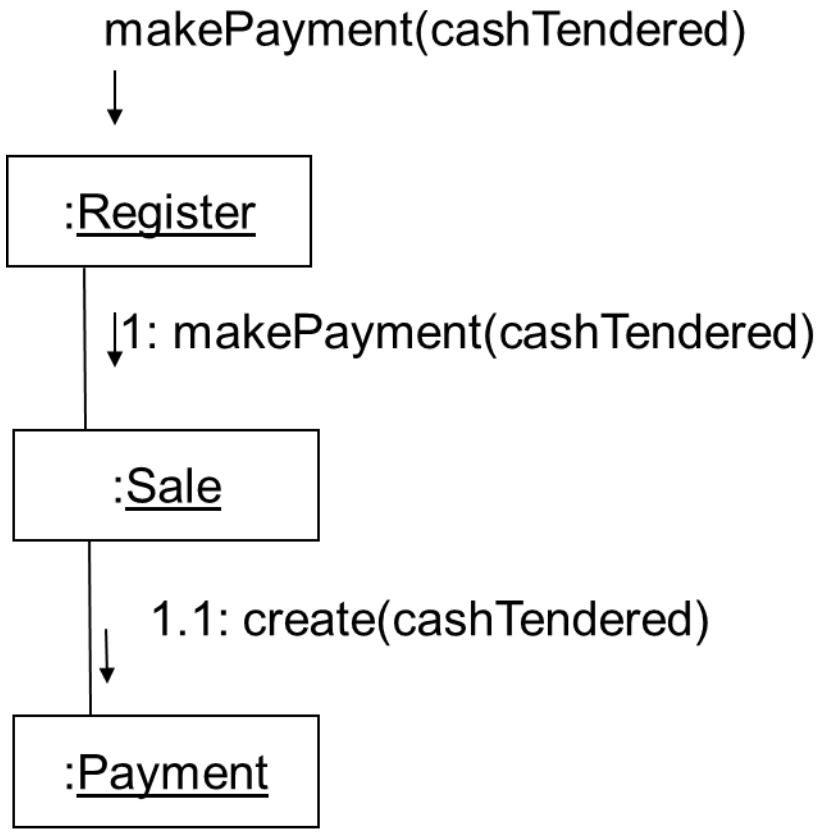
- Diagramas de Sequência ilustram interações em um formato tipo "cerca"
- O conjunto de todas as operações define o comportamento do sistema
- Será criado um diagrama de interação para cada contrato de operação



Exemplo de Diagrama de Colaboração: **efetuarPagamento**

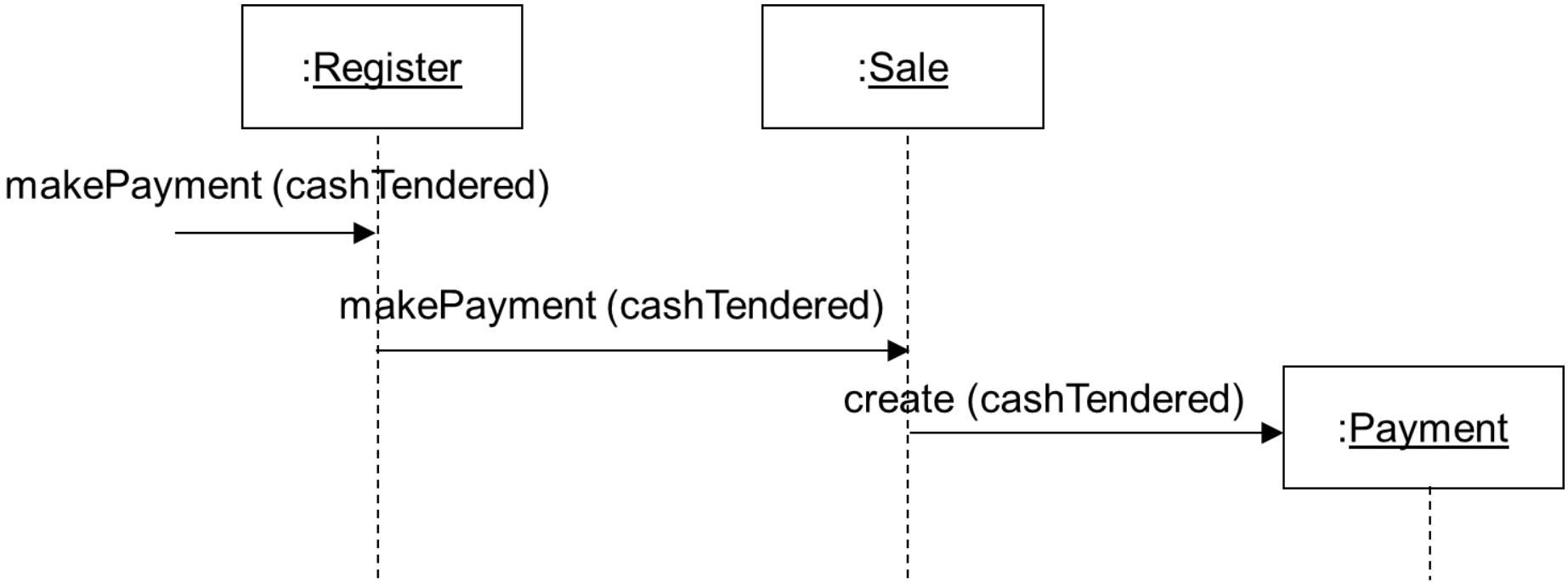


Como ler o Diagrama de Colaboração: efetuarPagamento

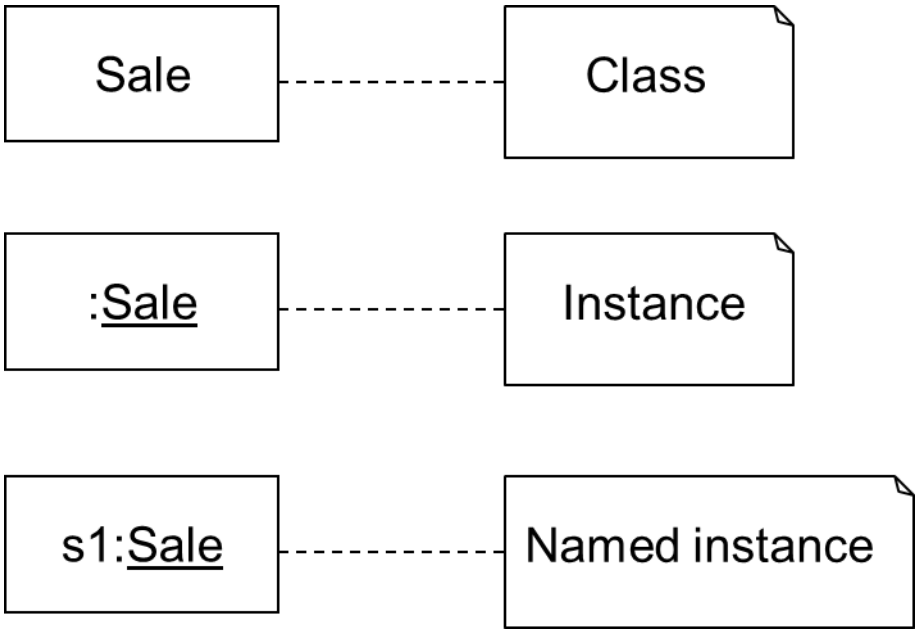


- 1. A mensagem **efetuarPagamento** é enviada para uma instância de Registradora. O emissor não é identificado.
- 2. A instância de Registradora envia uma mensagem **efetuarPagamento** para uma instância de Venda.
- 3. A instância de Venda cria uma instância de Pagamento

Exemplo de Diagrama de Sequência: efetuarPagamento



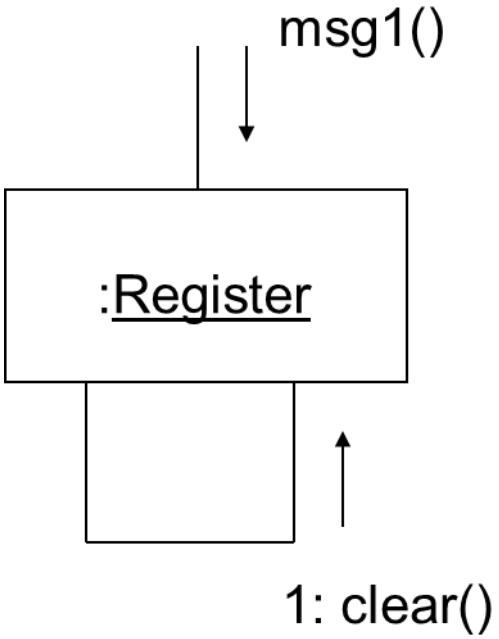
Ilustrando Classes e Instâncias



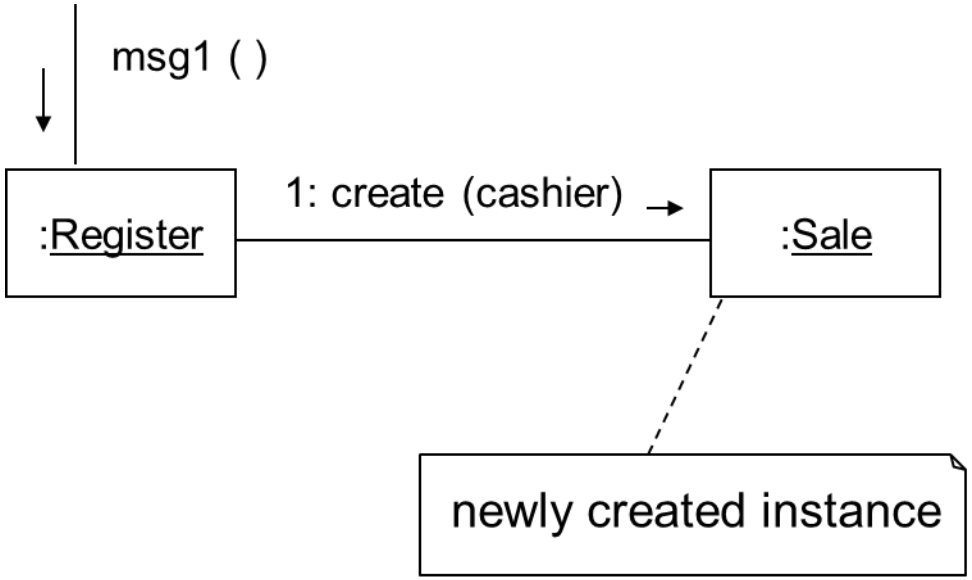
- Para mostrar uma instância de uma classe, usa-se o mesmo símbolo normal de classe, mas o nome é sublinhado.
- Adicionalmente, um nome de classe pode ser precedido por “:” (dois pontos).
- Um nome de instância pode ser usado para identificar unicamente uma instância

Mensagens para “self” ou “this”

- Uma mensagem pode ser enviada de um objeto para ele mesmo.
- Isto é ilustrado por um link para si mesmo, com mensagens fluindo ao longo do link

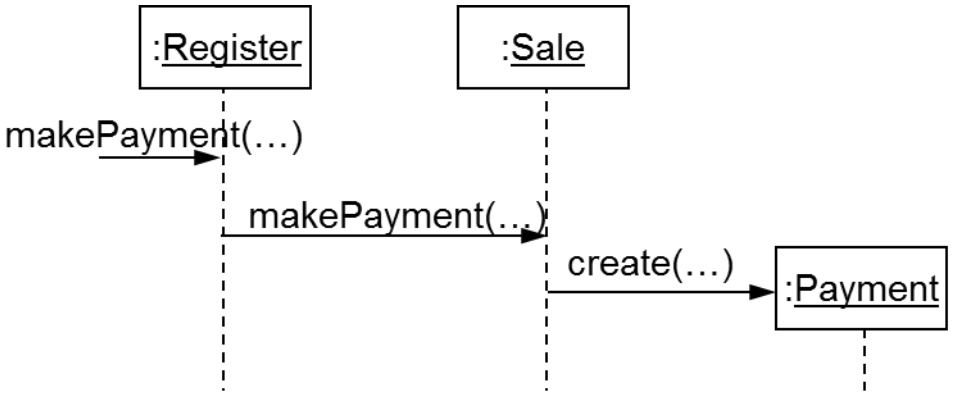


Criação de Instâncias



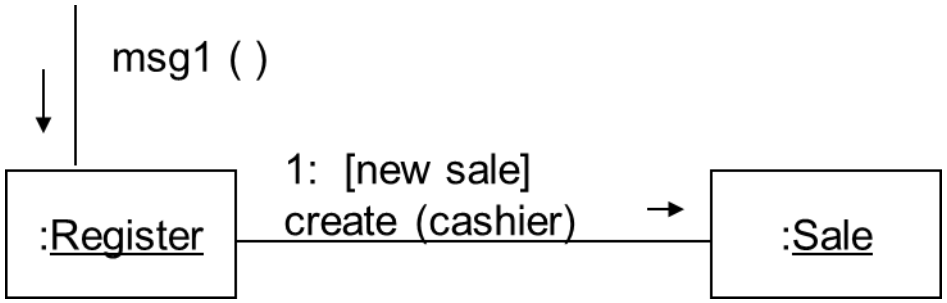
- A mensagem de criação de instâncias independente de linguagem é create, sendo enviada para a instância que está sendo criada.
- A mensagem create pode incluir parâmetros, indicando a passagem de valores iniciais

Criação de Instâncias



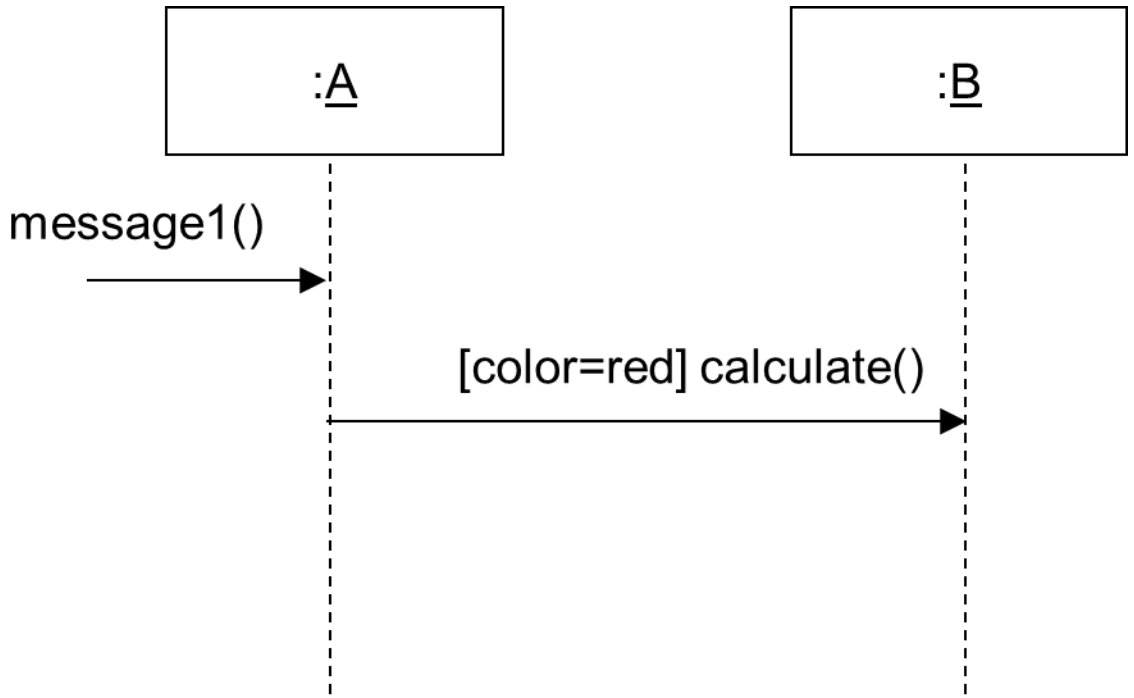
- Uma "linha da vida" de um objeto mostra a extensão da vida de um objeto no diagrama.
- Perceba-se que objetos novos são representados na altura da sua criação.

Mensagens condicionais

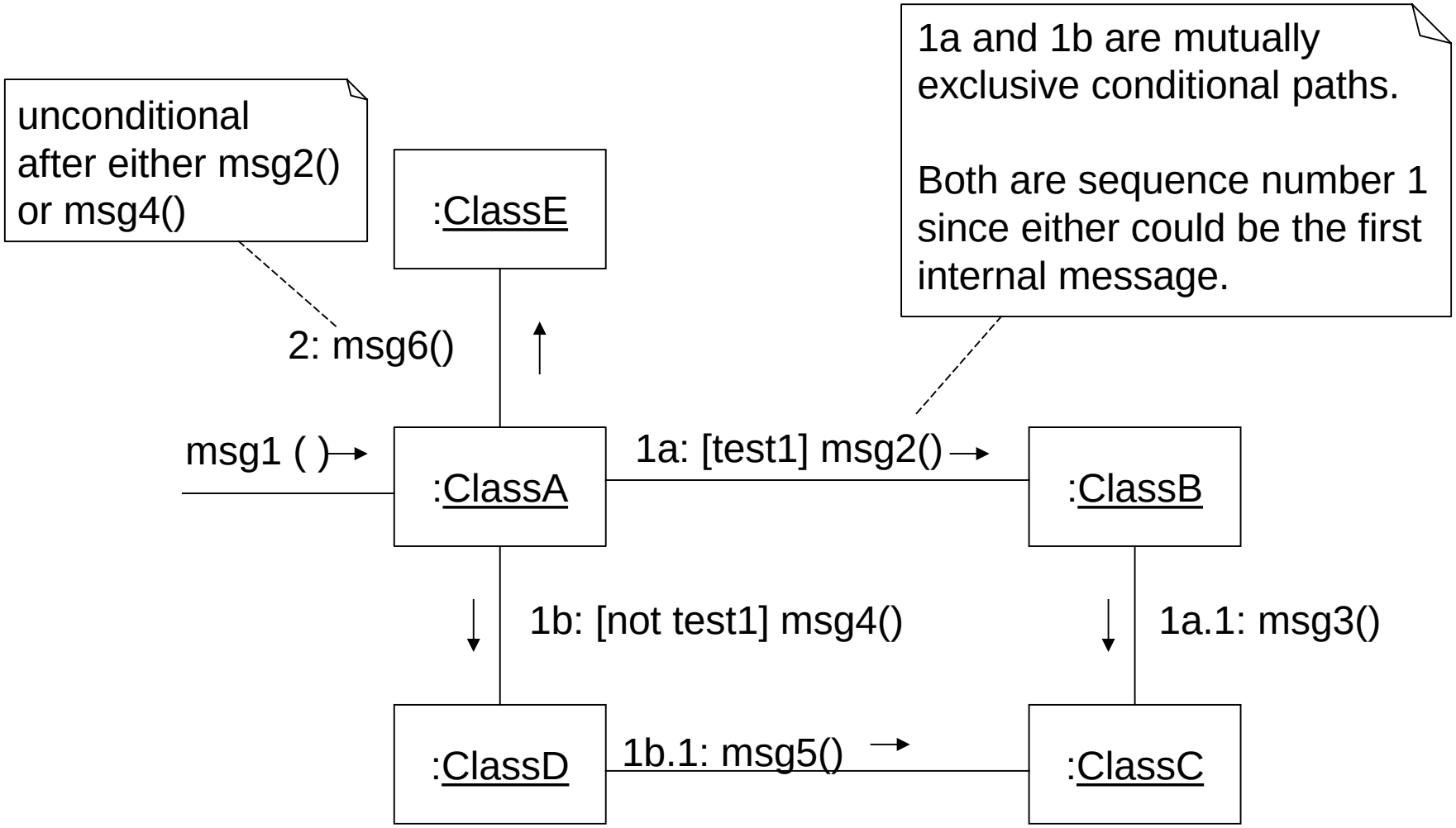


- Uma mensagem condicional é mostrada por uma cláusula condicional entre colchetes após o número de sequência, similar à cláusula de iteração.
- A mensagem é enviada somente se a cláusula for avaliada como verdadeira

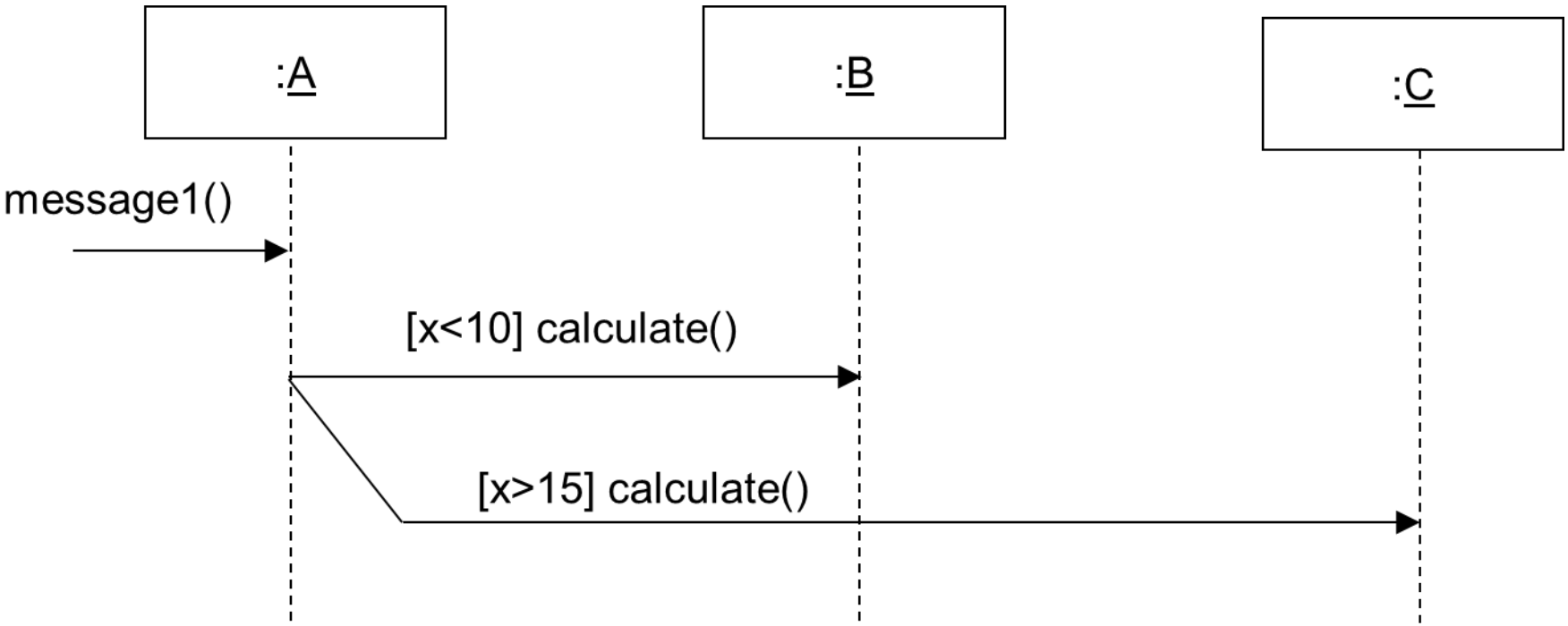
Mensagens condicionais



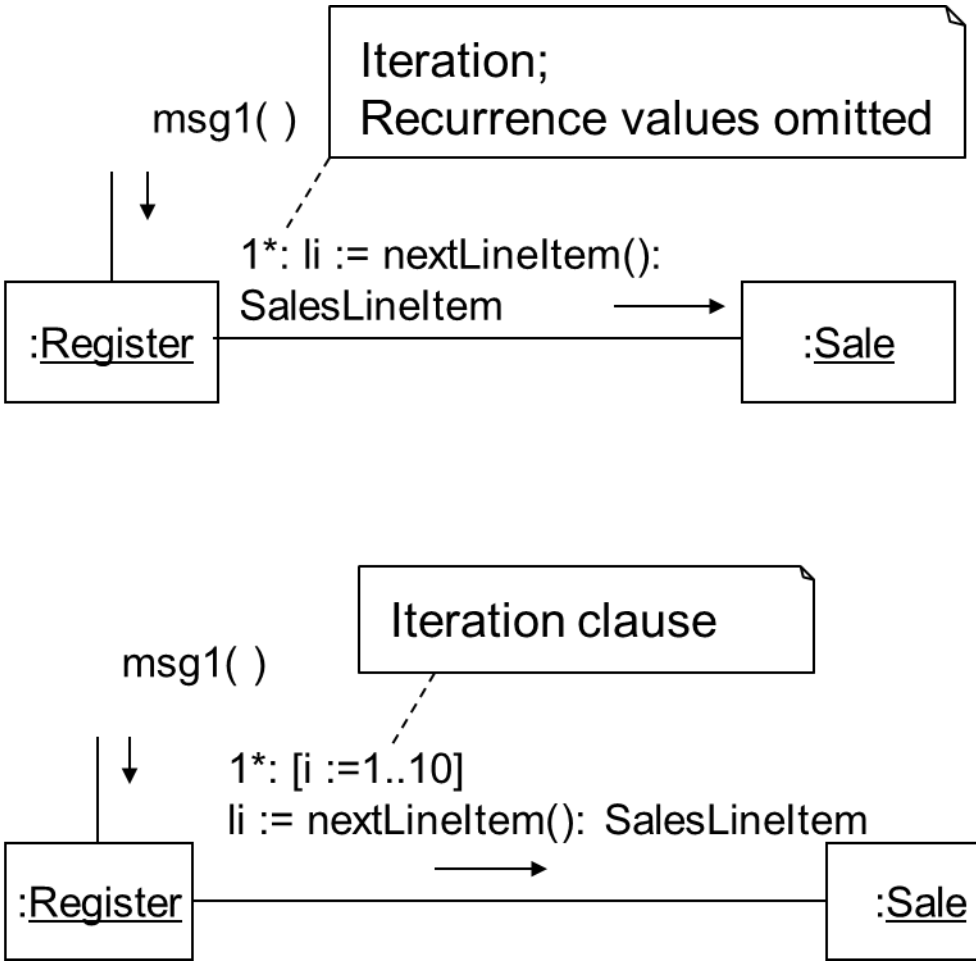
Caminhos condicionais mutuamente exclusivos



Mensagens condicionais mutuamente exclusivas



Iteração ou Looping



- Iteração é indicada por um "*" (asterisco) após o número de sequência.
- Isso expressa que a mensagem está sendo enviada repetidamente, em *looping*, para o receptor.
- Também é possível incluir uma cláusula de iteração, indicando os valores de recorrência

GRASP* : Projetando Objetos com Responsabilidades

*General Responsibility Assignment Software Patterns