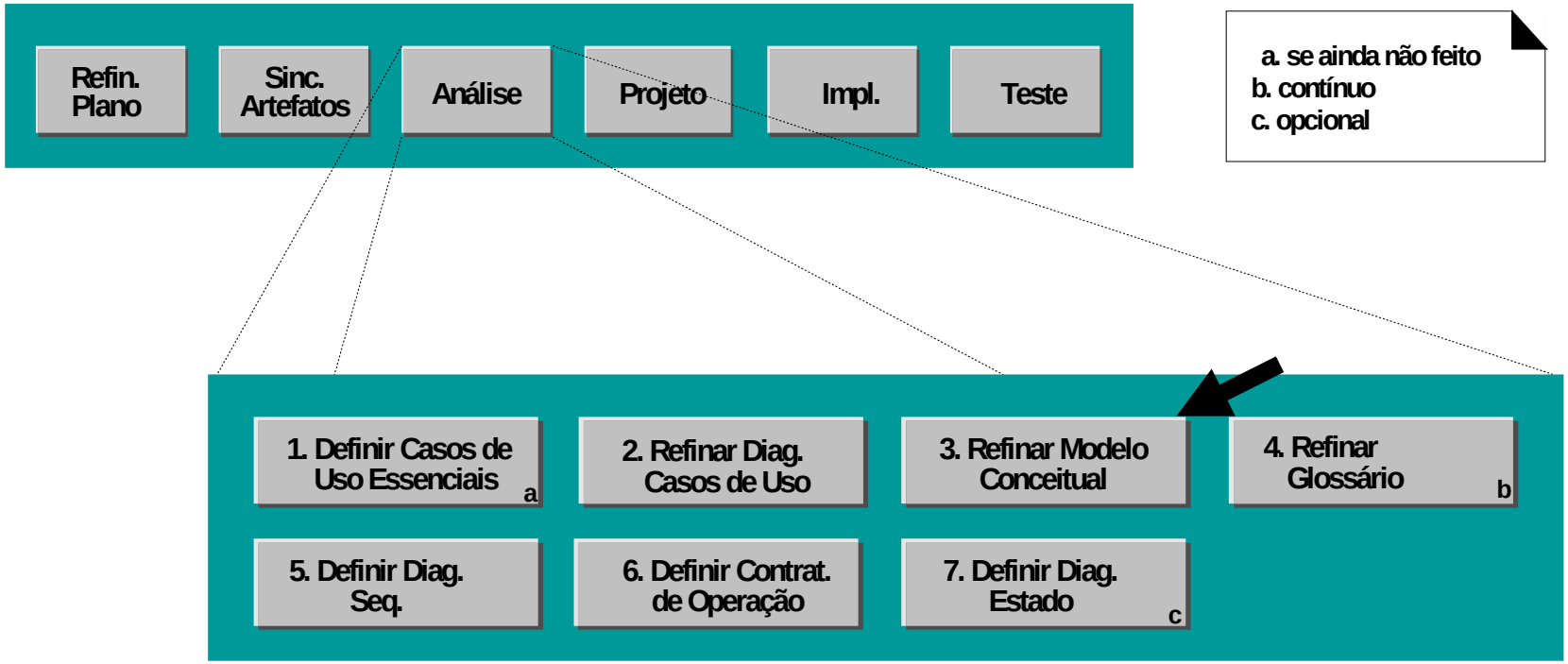
Análise e Projeto Orientados a Objeto
com UML e Padrões

Parte III

Análise (1)

Construindo um Modelo Conceitual

Um Ciclo de Desenvolvimento



Notas

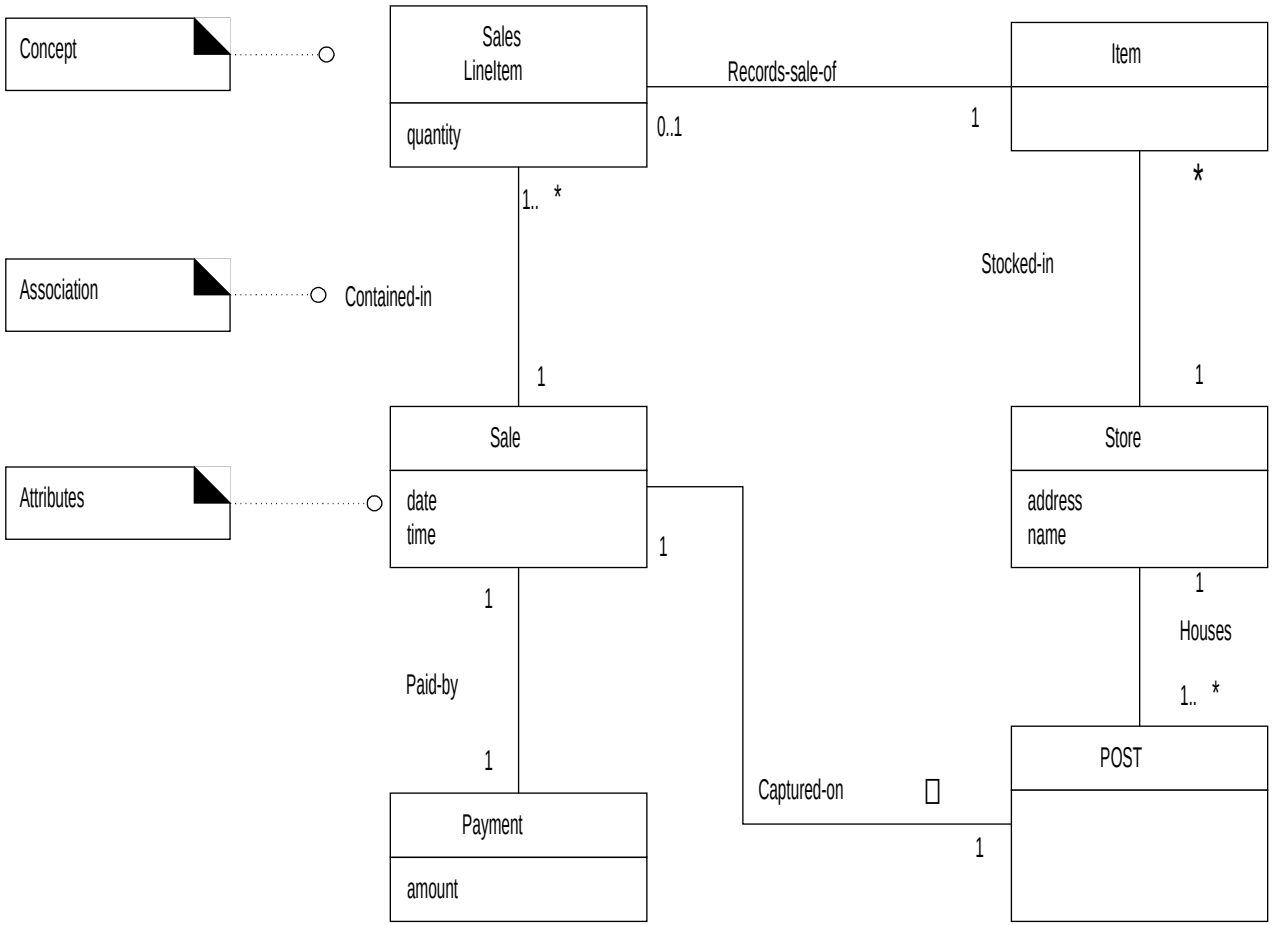
a. se ainda não feito
b. contínuo
c. opcional

Modelo Conceitual

- Artefato mais importante da AOO
- Representa conceitos relevantes (do ponto de vista do modelador) do domínio do problema
- Na UML, ilustrado com diagramas de estruturas estáticas contendo:
  - Conceitos
  - Associações entre conceitos
  - Atributos de conceitos

Modelo Conceitual para o Sistema POST

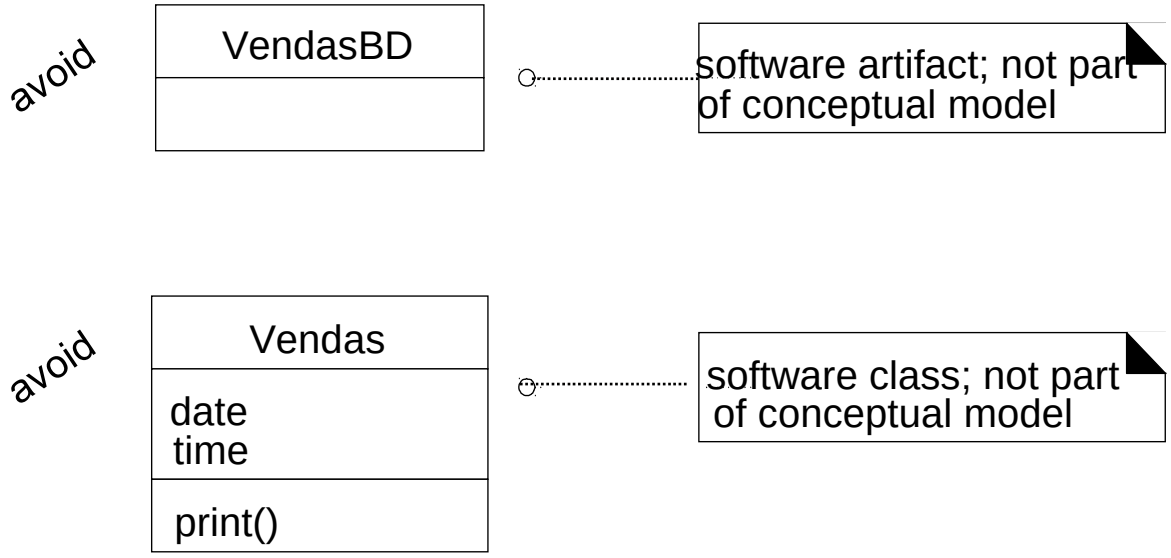
- Diagrama parcial



Conceitos

- **Idéias, coisas, ou objetos do mundo real**

- **<u>Não</u> representam componentes de software!**



Identificando Conceitos

- **Regras úteis:**
  - **É melhor especificar demais do que especificar de menos**
  - **Não exclua conceitos simplesmente porque os requisitos não indicam a necessidade de guardar informações sobre eles (comum em projeto de BD)**
  - **Comece fazendo uma lista de conceitos candidatos a partir de uma lista de conceitos comuns**
  - **Considere os substantivos e frases nominais nas descrições textuais do domínio do problema como possíveis candidatos a conceitos ou atributos**

Conceitos Típicos

Categoria	Exemplos
Objeto físico ou tangível	Terminal de ponto-de-venda Avião
Especificação, projeto, ou descrição de coisas	Especificação de produto Descrição de vôo
Lugares	Loja Aeroporto
Transações	Venda, Pagamento Reserva
Itens de transação	Itens de venda Parcelas de pagamento
Papéis de pessoas	Operador Piloto
Container de coisas	Loja Avião

Conceitos Típicos

Categoria	Exemplos
Coisas em um *container*	Item Passageiro
Sistemas externos	Serviço de crédito Controle de tráfego aéreo
Nomes abstratos	Fome Aracnofobia
Organizações	Departamento de vendas Companhia aérea
Eventos	Venda, Assalto, Reunião Vôo, Decolagem
Regras e políticas	Política de devolução Política de cancelamento

Conceitos Típicos

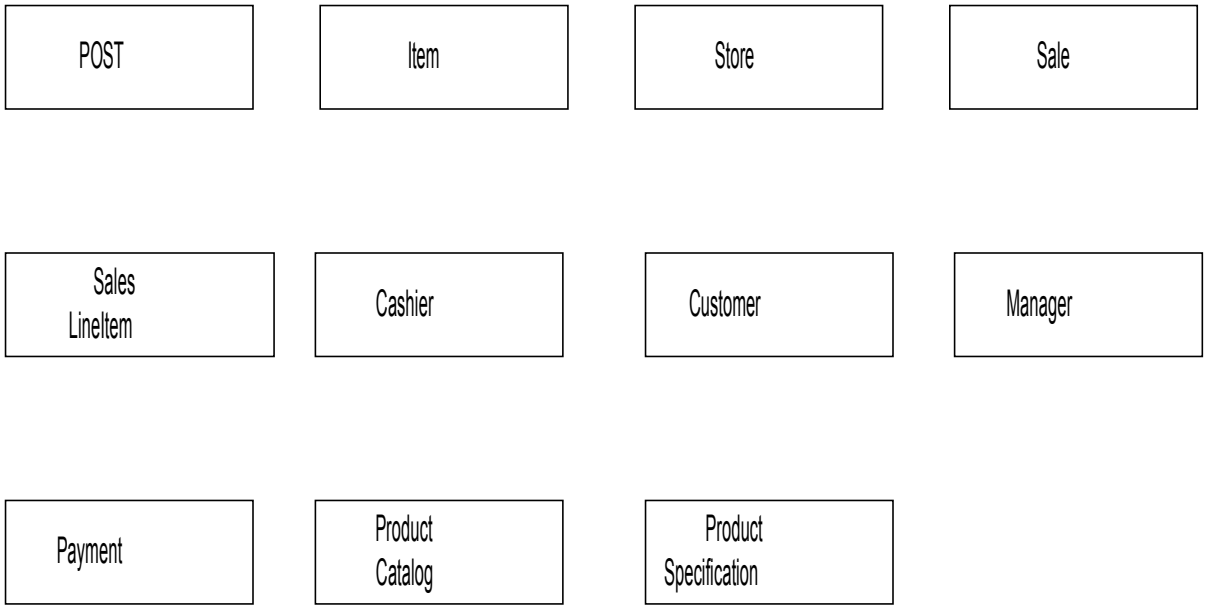
Categoria	Exemplos
Catálogos	Catálogo de produtos Catálogo de peças
Registros de finança, trabalho, contrato, questões legais	Recibo, Contrato de trabalho Registro de manutenção
Instrumentos e serviços financeiros	Linha de crédito Ações
Manuais, livros	Manual do empregado Manual de reparos

Identificando Conceitos e Atributos em Descrições Textuais

Ação do Ator	Resposta do Sistema
1. Este caso de uso começa quando um **Cliente** chega no **caixa** com **itens** para comprar.	
2. O **Operador** registra o **identificador** de cada **item**. Se há mais de um do mesmo **item**, o **Operador** também pode informar a **quantidade**.	**3.** Determina o **preço do item** e adiciona informação sobre o **item** à **transação de venda** em andamento. Mostra a **descrição** e o **preço** do **item** corrente.

- **Usar com cuidado!**
  - **Linguagens naturais: imprecisão e ambigüidade**

Conceitos Candidatos para o Sistema POST



- **Conceitos restritos ao caso de uso *Comprar Itens - Versão 1***

Conceitos de Relatório

- **Não incluir no modelo conceitual quando:**
  - **Toda informação contida no relatório é derivada de outras fontes**
- **Incluir no modelo conceitual quando:**
  - **Relatório tem um papel especial em termos das regras de negócio**

    Ex.: Recibo de venda dá direito à devolução dos itens comprados
- **Incluir *Recibo* no modelo conceitual para o sistema POST?**
  - **Sim, mas apenas no ciclo de desenvolvimento que trata do caso de uso *Devolver Itens***

Criando um Modelo Conceitual

- **Passos sugeridos:**
  1. **Liste os conceitos candidatos para os casos de usos em questão usando a lista de categorias comuns e identificação textual de nomes.**
  2. **Desenhe-os em um modelo conceitual.**
  3. **Adicione as associações necessárias para registrar os relacionamentos para os quais é preciso preservar alguma memória (*)**
  4. **Adicione os atributos necessários para cumprir os requisitos de informação. (*)**

(*) A ser discutido

Criando um Modelo Conceitual

- **Estratégia do "fazedor de mapas":**
  - **Usar nomes existentes no vocabulário do domínio**
  - **Incluir apenas conceitos pertinentes para os requisitos (casos de uso) em questão**
  - **Excluir tudo que não há no domínio do problema**

- **Erro comum: atributo em vez de conceito**

Vôo
destino

ou... ?

Vôo

Aeroporto
nome

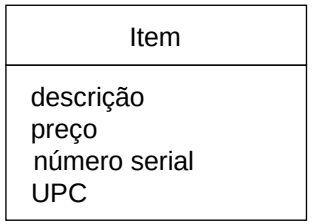
  - **Atributos normalmente correspondem a um texto ou número no mundo real**

Conceitos de Especificação ou Descrição

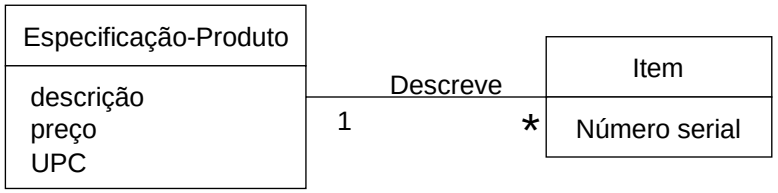
- **A especificação ou descrição de um objeto deve ser representada como um conceito em separado**
  - **evita perda de informação quando o objeto é deletado**
  - **reduz informações redundantes ou duplicadas**

- **Muito comum no domínio de produtos e vendas**

Ex.:



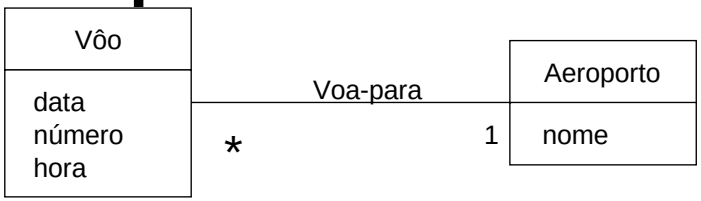
pior



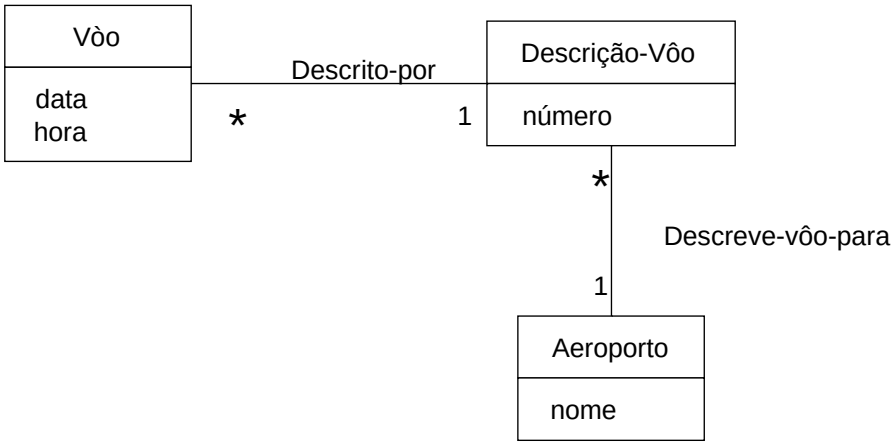
melhor

Conceitos de Especificação ou Descrição

- **Outro exemplo:**



pior



melhor

Conceitos e a Terminologia da UML

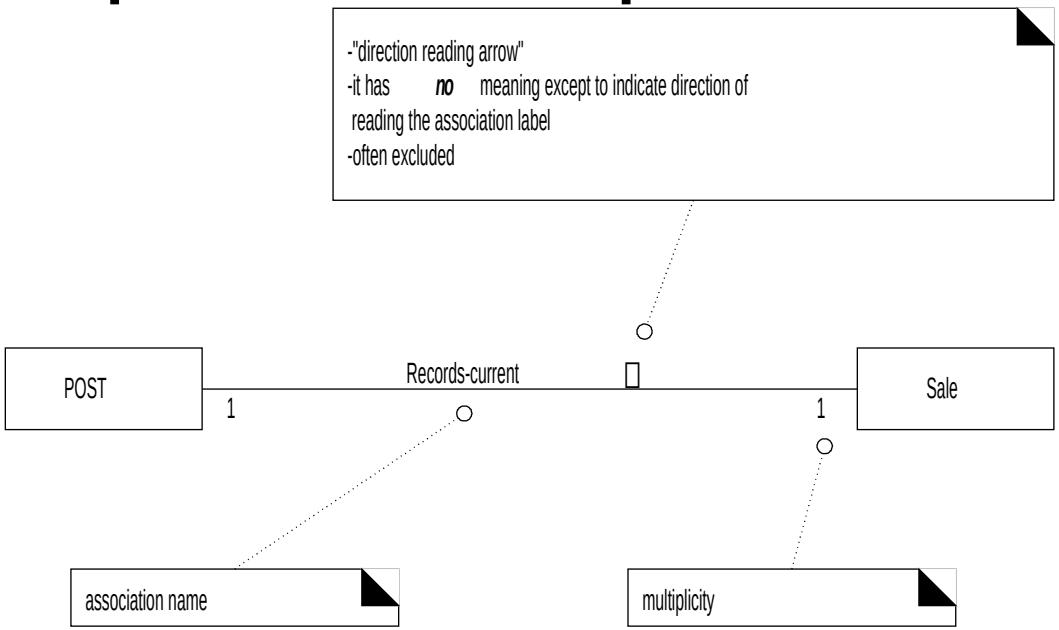
- UML usa o termo genérico “classe” para denotar tanto entidades do domínio da aplicação quanto classes na POO
  - Uma classe na POO é chamada mais especificamente de “classe de implementação”
- Os termos “tipo” e “interface” são usados para denotar *especificações* de classes de implementação
- No âmbito do curso, o termo “conceito” denota entidades do mundo real, e “classe” denota componentes de software e suas especificações

Associações

- Uma associação é um relacionamento entre conceitos que indica uma conexão significativa e interessante
- Descritos na UML como “relacionamentos estruturais entre objetos de tipos diferentes”
- Indicam conhecimento de um relacionamento que precisa ser preservado durante algum tempo
  - Duração de milisegundos ou anos, dependendo do contexto

Associações

- **Notação na UML**
  - **Uma linha entre dois conceitos mais um nome**
  - **Inerentemente bidirecional**
  - **Pode conter um seta de direção de leitura**
  - **Pontas podem conter expressões de cardinalidade**



Associações Típicas

Categoria	Exemplos
A é uma parte física de B (*)	Gaveta - POST Asa - Avião
A é uma parte lógica de B (*)	Item de Venda - Venda Escala - Vôo
A está fisicamente contido em B (*)	POST - Loja Passageiro - Avião
A está logicamente contido em B (*)	Descrição-Item - Catálogo Vôo - Roteiro de Viagem
A é uma descrição de B	Descrição-Item - Item Descrição-Vôo - Vôo
A é um item de uma transação ou relatório B	Item de Venda - Venda Opção de Reserva - Reserva
A é conhecido/registrado/reportado/capturado em B (*)	Venda - POST Reserva - Terminal de Reserva

(*) Alta prioridade

Associações Típicas

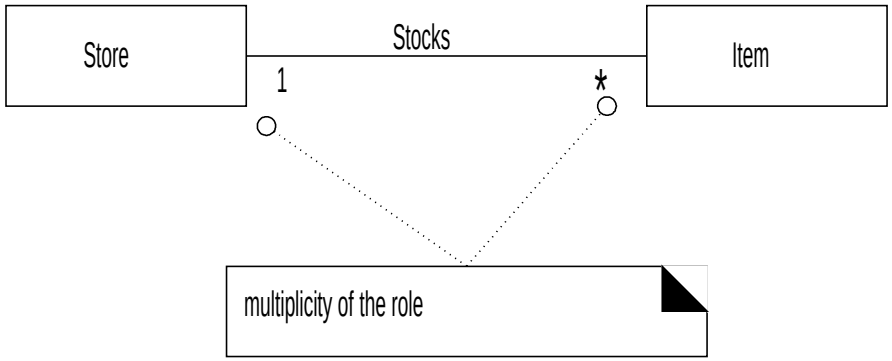
Categoria	Exemplos
A é um membro de B	Operador - Loja Piloto - Companhia Aérea
A é uma sub-unidade organizacional de B	Departamento - Loja Manutenção - Companhia Aérea
A usa ou gerencia B	Operador - POST Piloto - Avião
A se comunica com B	Cliente - Operador Agente de Reserva - Passageiro
A está relacionado com uma transação B	Cliente - Pagamento Passageiro - Bilhete
A é uma transação relacionada com outra transação B	Pagamento - Venda Reserva - Cancelamento
A é possuído por B	POST - Loja Avião - Companhia Aérea

Identificando Associações

- **Regras úteis:**
  - **Focar nas associações cujo conhecimento deve ser preservado**
  - **Usar nomes baseados em expressões verbais que façam sentido quando lidas no contexto do modelo**
  - **Evitar mostrar associações deriváveis ou redundantes**
  - **É mais importante identificar conceitos do que associações**
  - **Associações demais tendem a confundir um modelo conceitual ao invés de iluminá-lo**

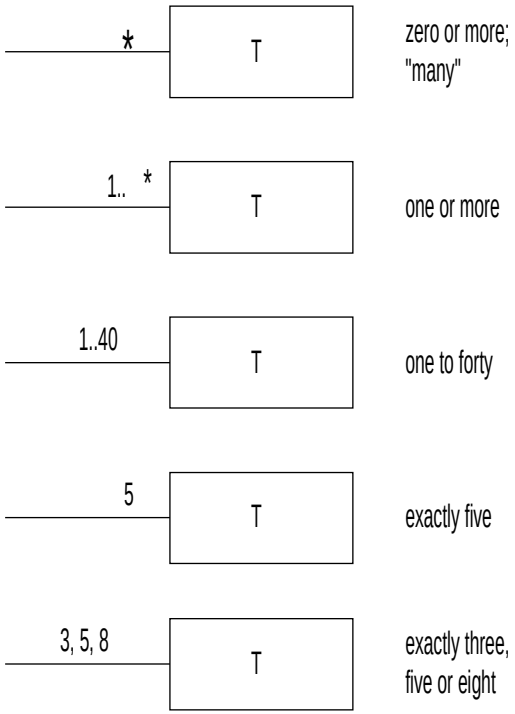
Papeis de Uma Associação

- Cada ponta de um associação é chamada de um "papel"
- Um papel pode ter:
  - Nome
  - Expressão de multiplicidade



  - Navegabilidade

Expressões de Multiplicidade



- **O valor da multiplicidade depende do contexto**
  - **Ex.: Pessoa *Trabalha-para* Empresa**

Adicionando Associações ao Modelo Conceitual do Sistema POST

- **Relacionamentos fundamentais**
  - **Venda *Capturada-em* POST**

    Para conhecer a venda corrente, calcular total e imprimir recibo
  - **Venda *Paga-por* Cliente**

    Para saber se a venda foi paga, calcular troco, e imprimir recibo
  - **Catálogo-Produto *Contém* Especificação-Item**

    Para obter a especificação de um item, dado um UPC

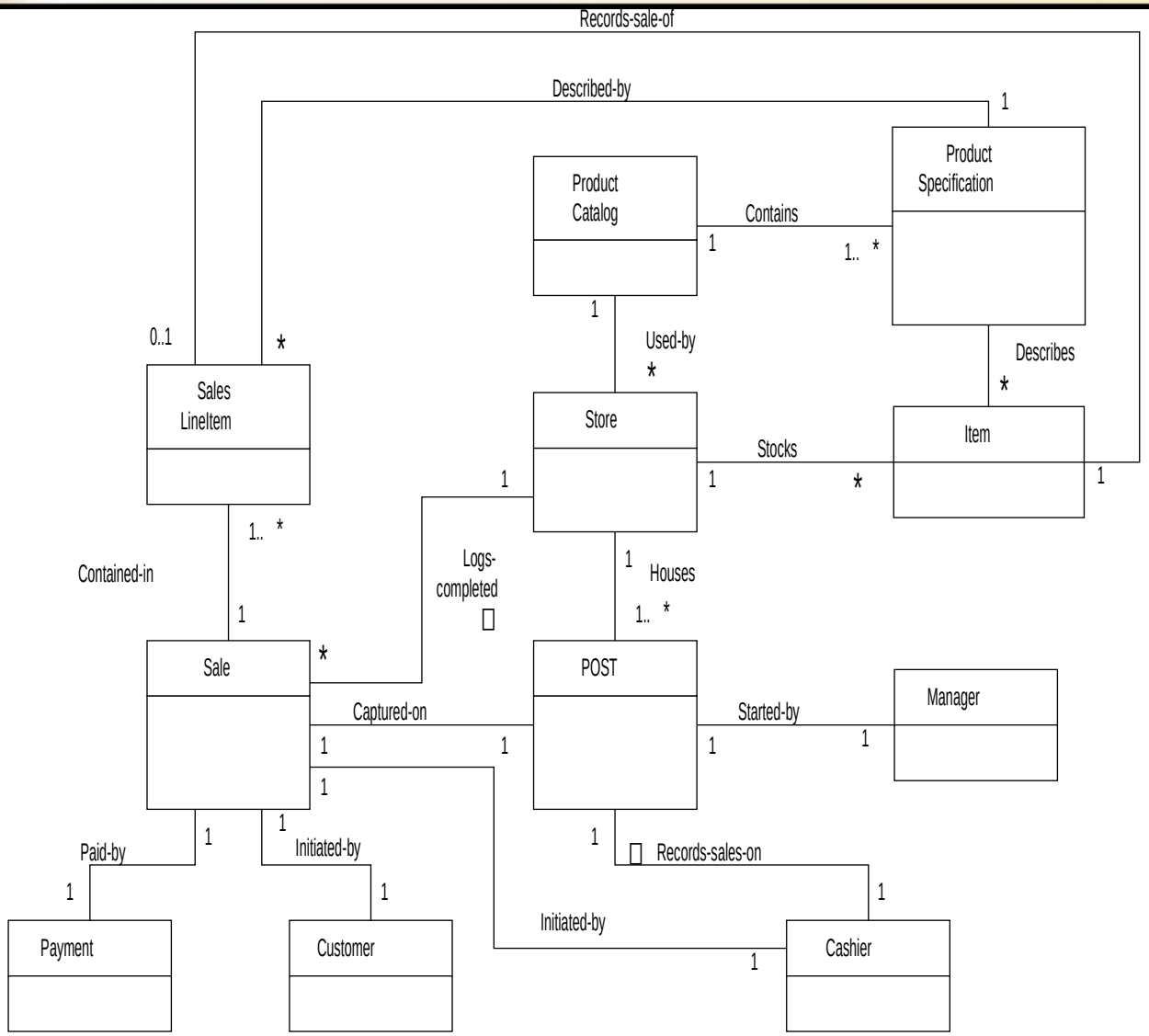
Aplicando a Lista de Associações Típicas

Categoria	Exemplos
A é uma parte física de B	N.A.
A é uma parte lógica de B	Item de Venda - Venda
A está fisicamente contido em B	POST - Loja Item - Loja
A está logicamente contido em B	Especificação-Produto - Catálogo Catálogo - Loja
A é uma descrição de B	Especificação-Produto - Item
A é um item de uma transação ou relatório B	Item de Venda - Venda
A é conhecido/registrado/reportado/capturado em B	Venda (corrente) - POST Venda (completada) - Loja

Aplicando a Lista de Associações Típicas

Categoria	Exemplos
A é um membro de B	Operador - Loja
A é uma sub-unidade organizacional de B	N.A.
A usa ou gerencia B	Operador - POST Gerente - POST
A se comunica com B	Cliente - Operador
A está relacionado com uma transação B	Cliente - Pagamento Operador - Pagamento
A é uma transação relacionada com outra transação B	Pagamento - Venda
A é possuído por B	POST - Loja

Conceitos e Associações Candidatos para o Sistema POST



Eliminando Associações Redundantes ou Desnecessárias

- **Associações cujo conhecimento não precisa ser preservado podem ser removidas do modelo:**

Associação	Consideração
Venda *Iniciada-por* Operador	Conhecimento não exigido nos requisitos; derivável da associação Operador *Registra-vendas-em* POST.
Operador *Registra-vendas-em* POST	Conhecimento não exigido nos requisitos.
POST *Inicializado-por* Gerente	Conhecimento não exigido nos requisitos.
Venda *Iniciada-por* Cliente	Conhecimento não exigido nos requisitos.
Loja *Armazena* Item	Conhecimento não exigido nos requisitos.
Item de Venda *Registra-venda-de* Item	Conhecimento não exigido nos requisitos.

Preservando Associações de Compreensão

- **Preservar apenas associações de conhecimento pode resultar num modelo que não transmite um completo entendimento do domínio**
  - **Ex.: Venda *Iniciada-por* Cliente**

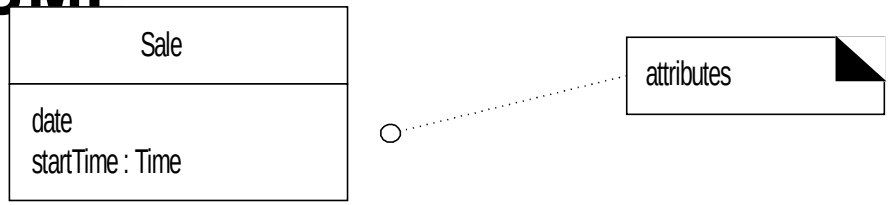
    Remoção deixa de fora um aspecto importante do domínio— o fato de que um cliente gera uma venda

- **Modelo conceitual é um artefato de comunicação!**

- **Regra geral:**
  - **Enfatizar associações de conhecimento, mas preservar associações que enriquecem o entendimento do domínio**

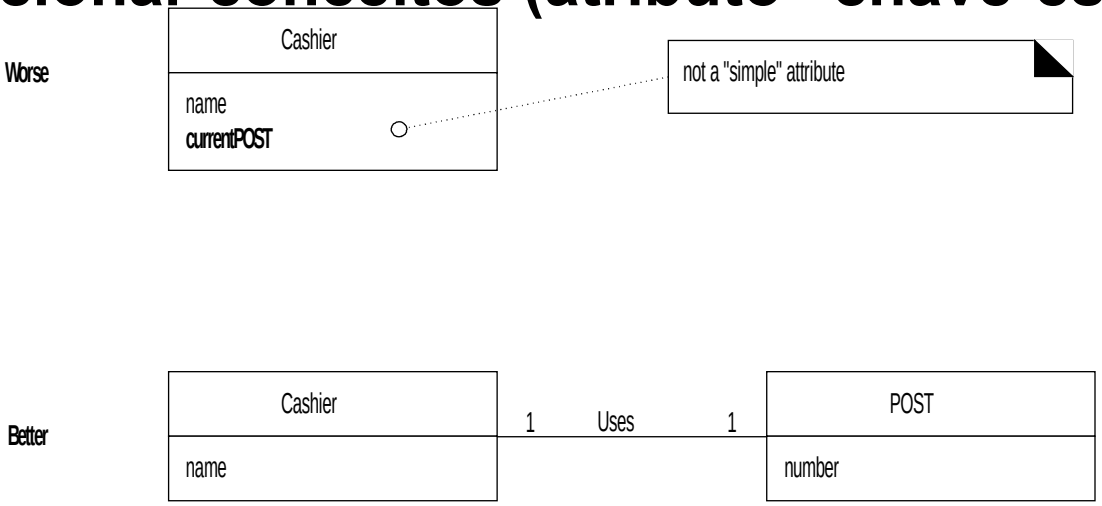
Atributos

- Um atributo é um dado lógico de um objeto do domínio
- Definidos para conceitos cujos requisitos (casos de uso) sugerem a necessidade de preservar algum tipo de informação
  - Ex.: atributos *data* e *hora* para o conceito Venda
- Notação na UML



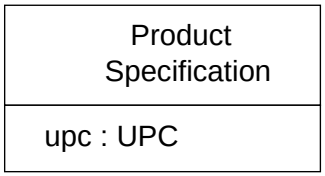
Identificando Atributos

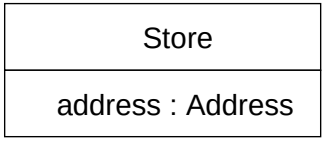
- **Atributos devem preferencialmente representar tipos primitivos de dados ou de valores simples**
  - **Ex.: *Data*, *Número*, *Texto*, *Hora*, *Endereço*, etc.**
- **Atributos <u>não</u> devem ser usados para:**
  - **Representar um conceito complexo**
  - **Relacionar conceitos (atributo "chave-estrangeira")**



Atributos de Tipo Não-Primitivo

- **Podem ser representados diretamente no modelo conceitual**





- **Um atributo deve ser de tipo não-primitivo quando:**
  - **É composto de seções separadas**

    Ex.: endereço, data
  - **Precisa ser analisado ou validado**

    Ex.: CPF, número de matrícula
  - **Possui outros atributos**

    Ex.: Um preço promocional com prazo de validade
  - **É uma quantidade com uma unidade**

    Ex.: valores monetários, medidas

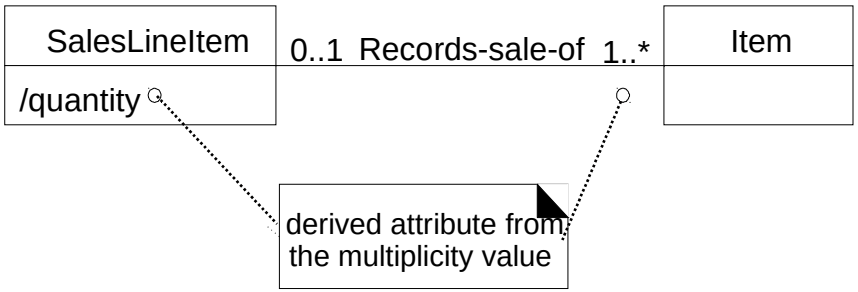
Adicionando Atributos aos Conceitos Candidatos do Sistema POST

Conceito	Atributos e justificativa
Pagamento	*quantia*—Para determinar se pagamento é suficiente e calcular troco.
Especificação-Produto	*descrição*—Para mostrar na tela e imprimir no recibo. *UCP*—Para localizar especificação do item. *preço*—Para calcular o total da venda.
Venda	*data*, *hora*—Para imprimir no recibo e registrar no log de vendas.
Item de Venda	*quantidade*—Para registrar a quantidade digitada quando há mais de um do mesmo item.
Loja	*nome*, *endereço*—Para imprimir no recibo.

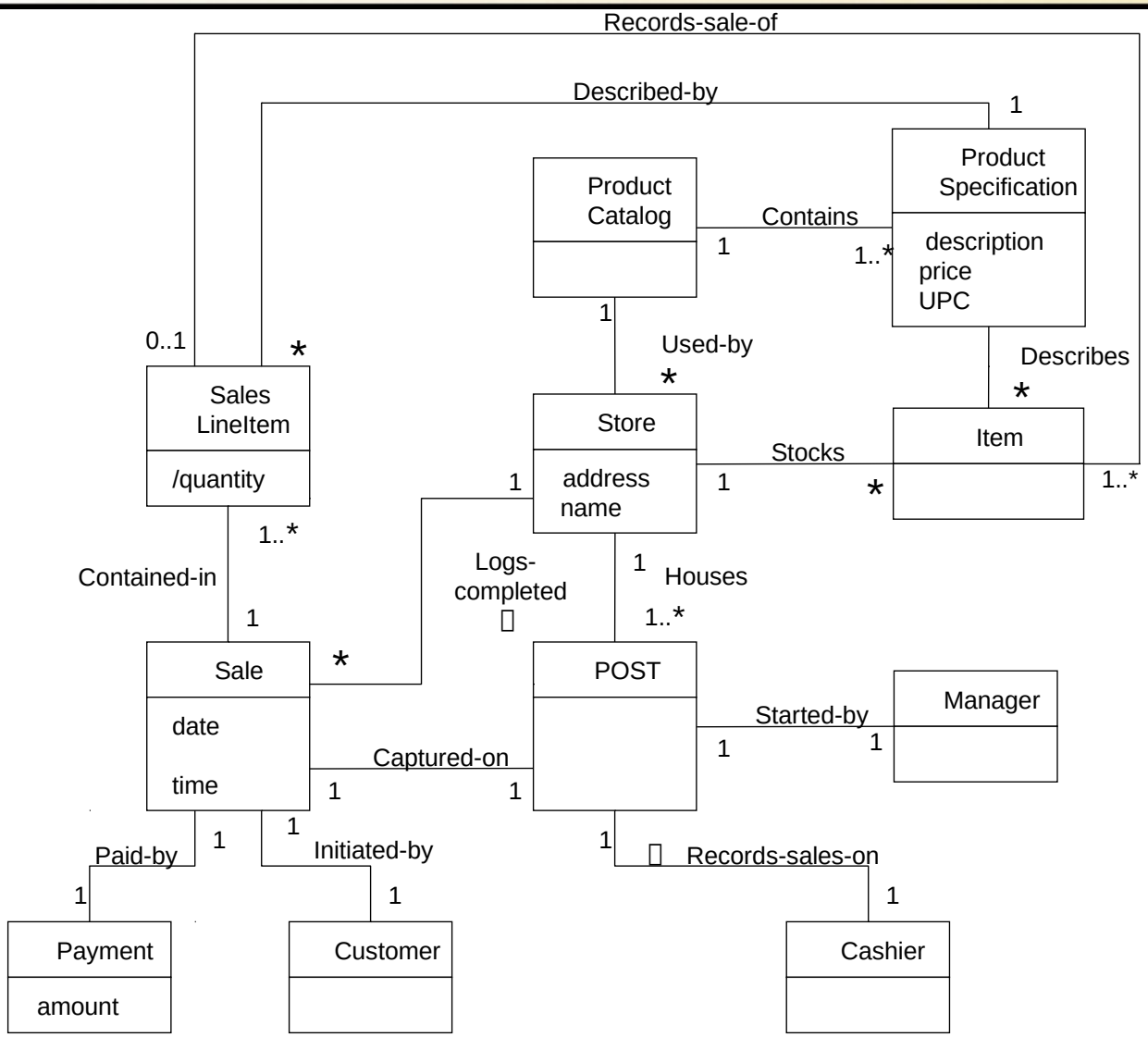
Atributo Derivado

- **Um atributo *"derivado"* é um atributo cujo valor pode ser deduzido a partir de outras informações**
  - **Ex.: *quantidade* em *Item de Venda*—pode ser deduzido a partir da multiplicidade da associação entre *Item de Venda* e *Item***

- **Na UML, indicado com o símbolo "/"**



Modelo Conceitual Inicial do Sistema POST



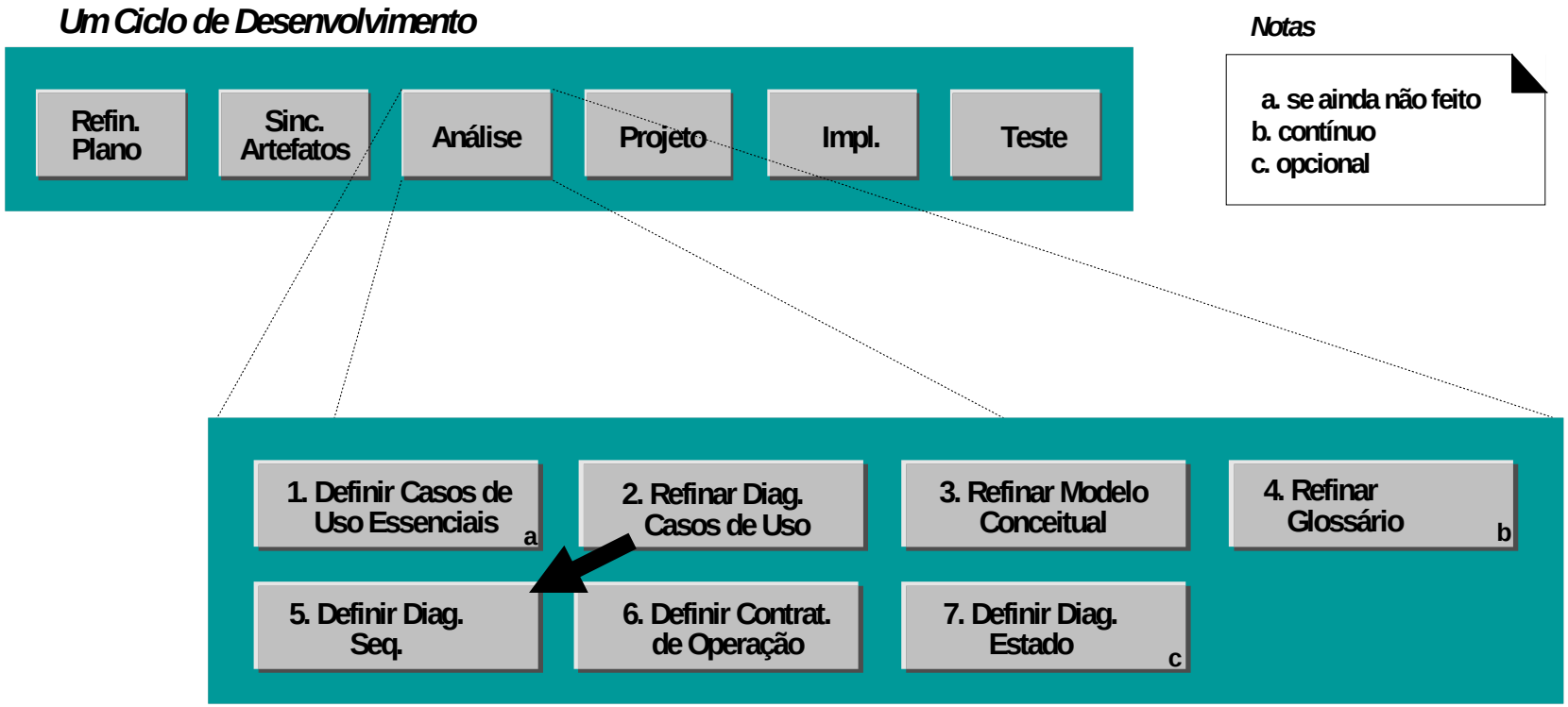
Registrando Termos no Glossário

- Um *glossário* ou *dicionário de modelo* é um documento que define os termos (ou vocabulário) do domínio
  - Similar a um dicionário de dados usado na modelagem de BD
- Fundamental para garantir uma comunicação consistente e um entendimento compartilhado entre usuários e desenvolvedores
- Também pode ser usado para registrar restrições de domínio e regras de negócio (não explorado no curso)

Glossário Parcial para o Sistema POST

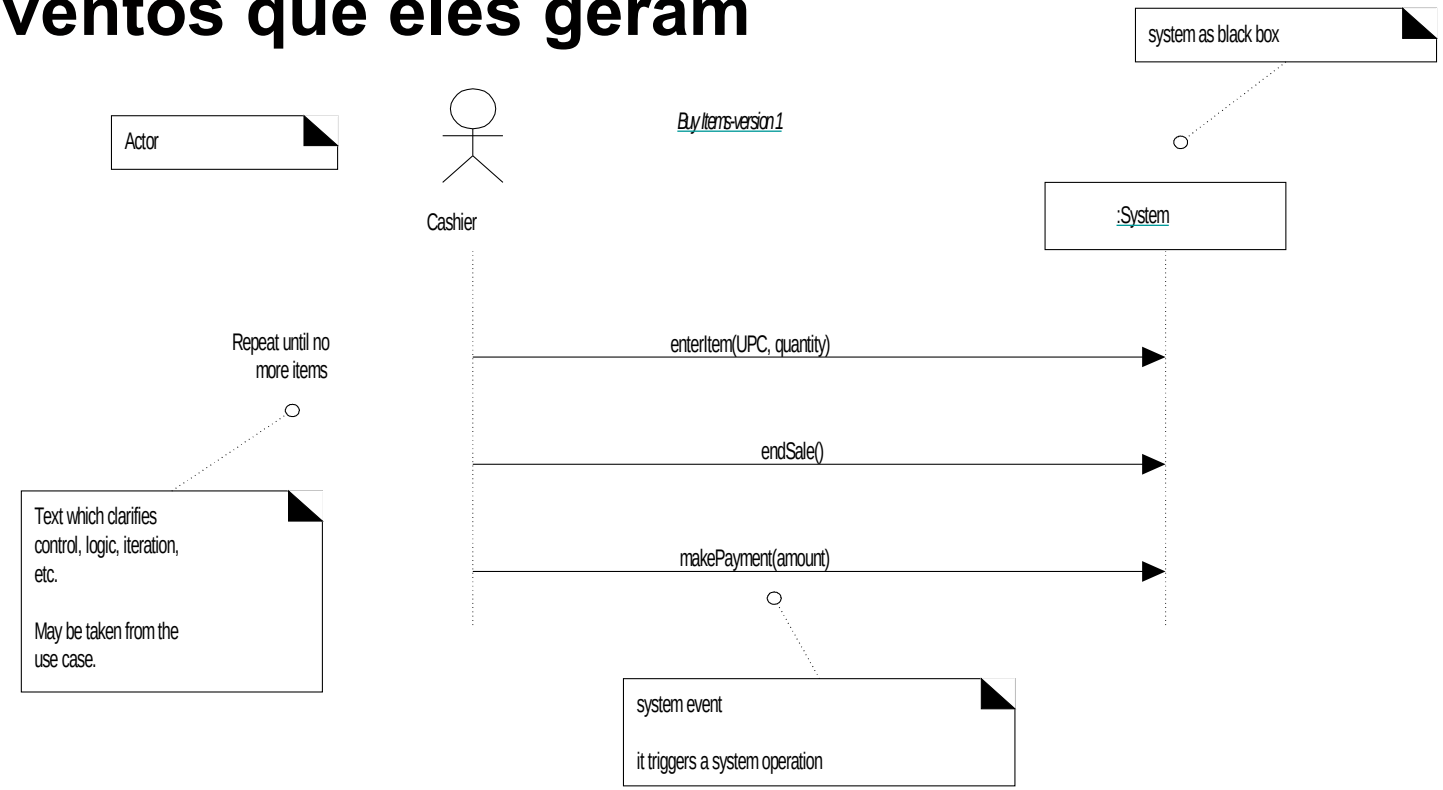
Termo	Categoria	Comentário
Comprar Itens	caso de uso	Descrição do processo de um cliente comprando itens numa loja.
Especificação-Produto.descrição :Texto	atributo	Uma descrição sucinta de um item de venda.
Item	tipo	Um item à venda numa loja.
Pagamento	tipo	Um pagamento em dinheiro.
Especificação-Produto.preço :Quantidade	atributo	O preço de um item de venda.
Item de Venda.quantidade :Inteiro	atributo	A quantidade de um tipo particular de item comprado.
Venda	tipo	Uma transação de venda.
Item de Venda	tipo	Um item particular comprado como parte de uma venda.

Definindo Diagramas de Seqüência



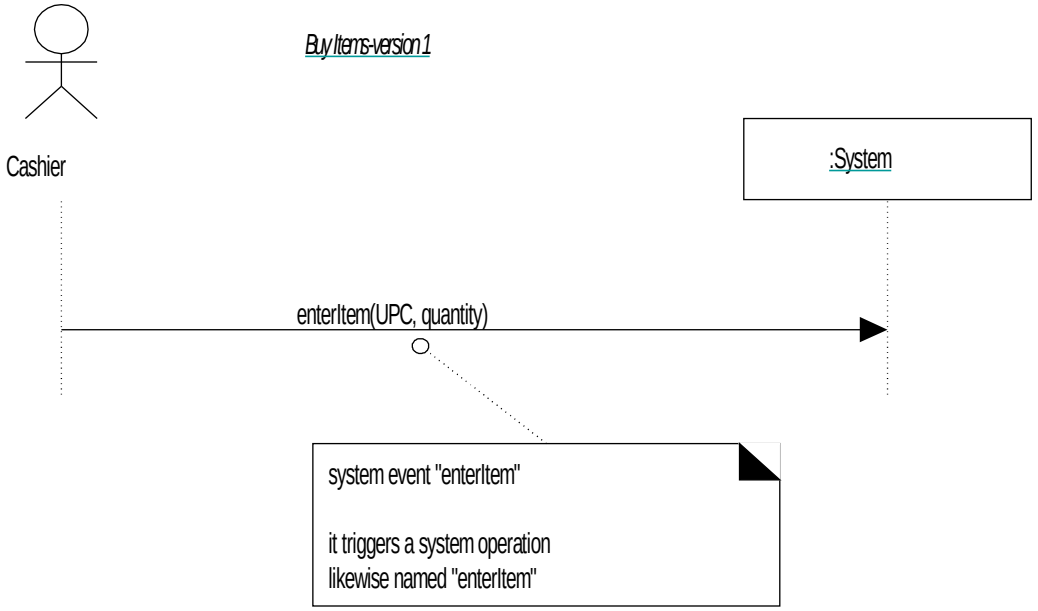
Diagramas de Seqüência

- Um *diagrama de seqüência* ilustra a ordem das interações dos atores externos com o sistema (representado como uma “caixa-preta”) e os eventos que eles geram



Eventos e Operações

- Um *evento de sistema* é um evento externo de entrada gerado por um ator do sistema
  - Inicia uma operação de resposta de mesmo nome
- Uma *operação de sistema* é uma operação do sistema que executa em resposta a um evento de sistema



Representando Operações

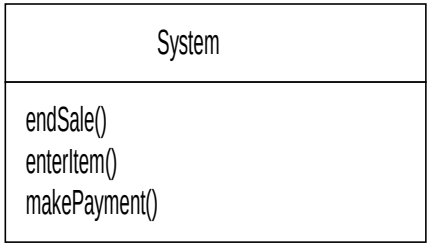
- **O conjunto necessário de operações de sistema é determinado através da identificação dos eventos de sistema**
  - **Exemplos de operações para o sistema POST:**

    entrarItem(UPC, quantidade)

    encerrarVenda()

    fazerPagamento(quantia)

- **Na UML, representado como operações de um tipo denominado *Sistema*:**

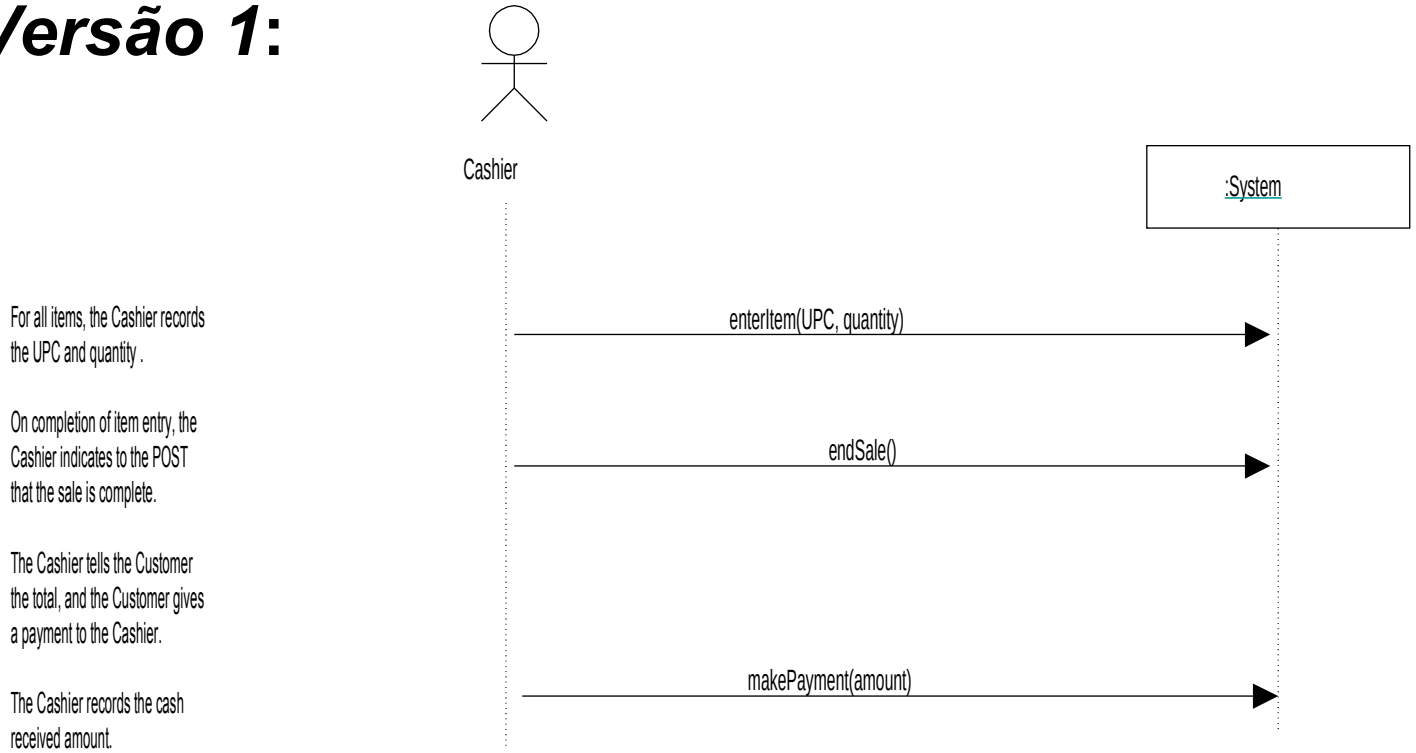


Definindo Diagramas de Seqüência

- **Regras úteis:**
  1. **Desenhar uma linha vertical representando o sistema como uma caixa-preta.**
  2. **Identificar os atores que operam diretamente com o sistema. Desenhar uma linha vertical representando cada um desses atores.**
  3. **A partir da descrição das seqüências típicas de eventos dos casos de uso, identificar os eventos de sistema que cada ator gera. Ilustrar os eventos no diagrama.**
  4. **Opcionalmente, incluir o texto do caso de uso à esquerda do diagrama.**

Definindo Diagramas de Seqüência

- **Diagrama de seqüência para o sistema POST com (parte do) texto do caso de uso *Compra Itens -Versão 1*:**

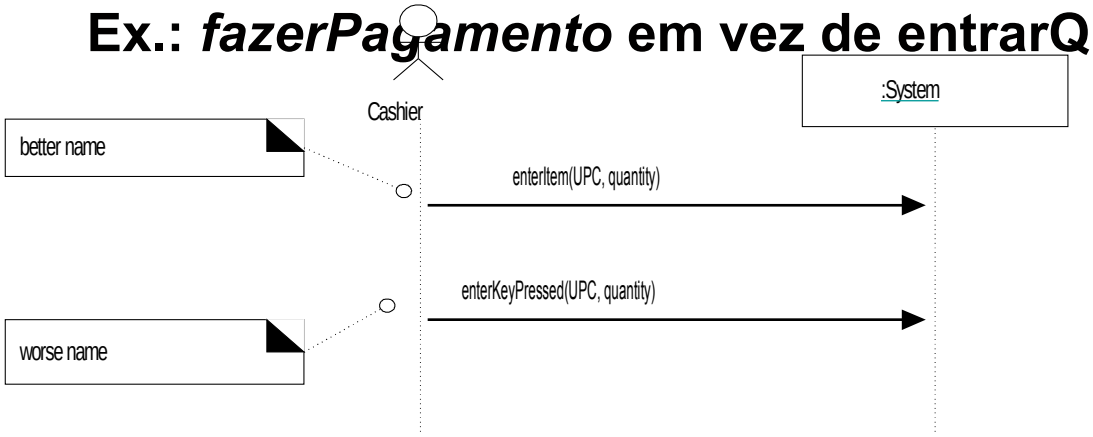


Nomeando Eventos e Operações

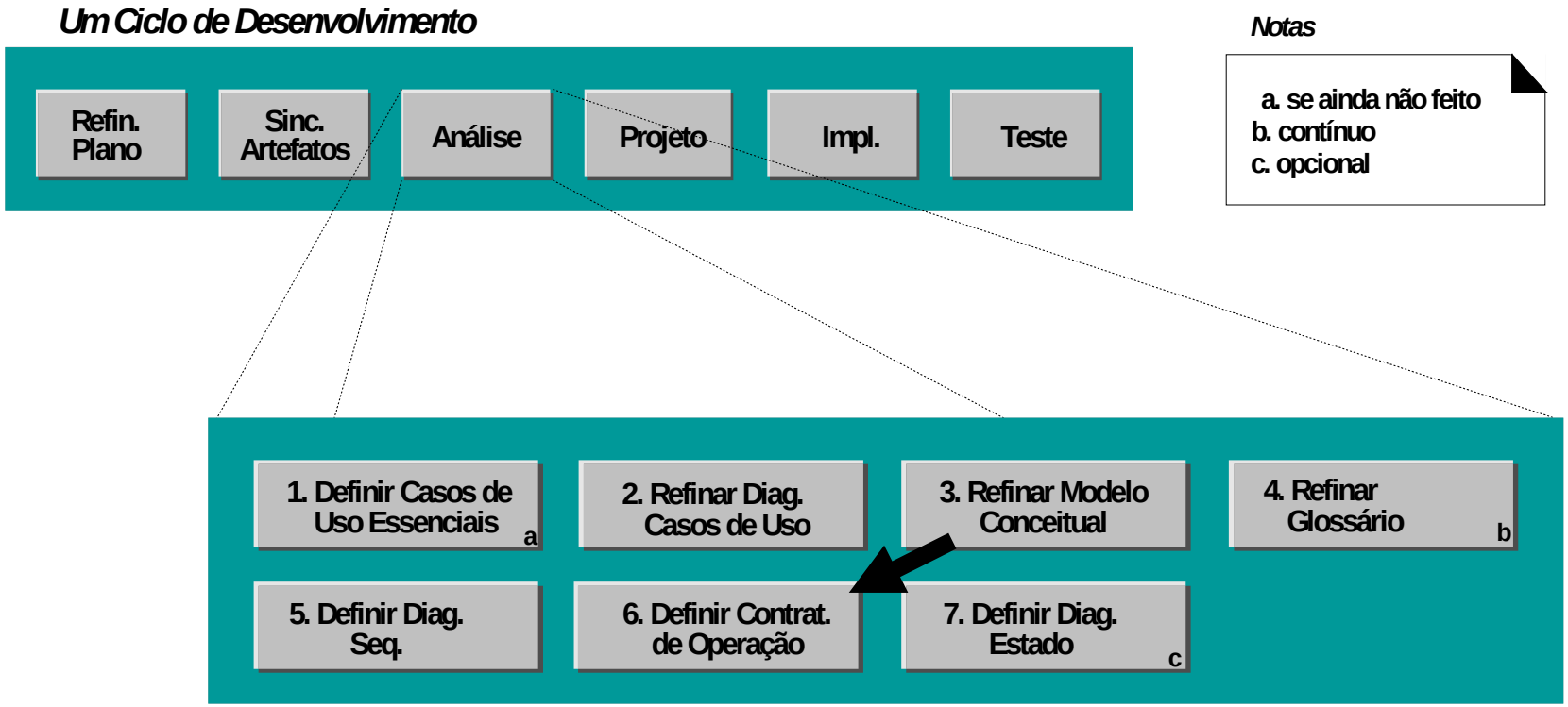
- **Regras úteis:**
  - **Começar com um verbo**
  - **Enfatizar "*intenção*" em vez do meio físico de entrada ou componente gráfico da interface com o usuário**

    Ex.: *encerrarVenda* em vez de *pressionarTeclaEnter*
  - **Expressar intenção no nível mais alto de abstração**

    Ex.: *fazerPagamento* em vez de entrarQuantia



Definindo Contratos de Operação



Contratos

- **Um *contrato* é um documento que descreve os compromissos de uma operação**
  - **Estilo declarativo**
  - **Pré e pós-condições de mudanças de estado**
  - **Para métodos, classes, ou operações gerais de sistema**
- **Exemplo para operação *entrarItem*:**

Contrato:	entrarItem (upc :número, quantidade :inteiro)
Responsabilidades:	Registra venda de um item e o adiciona à venda corrente. Mostra descrição e preço do item.
Tipo:	Sistema
Referencia:	Funções: R1.1, R1.3, R1.9
	Casos de uso: Comprar Itens

Contratos

- **Exemplo para operação *entrarItem* (cont.):**

Notas: Usar acesso rápido ao BD

Exceções: Se UPC inválido, indicar erro.

Saída:

Pré-condições: UPC é conhecido do sistema

Pós-condições:

- Se nova venda, uma *Venda* foi criada (criação de instância).
- Se nova venda, a nova *Venda* foi associada com um *POST* (formação de associação.
- Um *Item-de-Venda* foi criado (criação de instância).
- O *Item-de-Venda* foi associado à *Venda* (formação de associação).
- *Item-de-Venda.quantidade* foi definido para *quantidade* (modificação de atributo).
- O *Item-de-Venda* foi associado com uma *Especificação-Produto*, baseado no casamento de UPCs (formação de associação).

Seções de um Contrato

Contrato:	Nome e parâmetros da operação
Responsabilidades:	Descrição informal das responsabilidades da operação
Tipo:	Nome do tipo (conceito, classe, interface)
Referencia:	Funções, casos de uso, etc.
Notas:	Notas de projeto, algoritmos, etc.
Exceções:	Casos excepcionais
Saída:	Saídas não-IU, tais como mensagens ou registros enviados para fora do sistema
Pré-condições:	Pré-suposições sobre o estado do sistema antes da execução da operação

Pós-condições:

- O estado do sistema após a execução da operação

Como Fazer um Contrato

- **Regras úteis:**

1. **Identificar operações de sistema a partir dos diagramas de seqüência.**
2. **Para cada operação, construir um contrato.**
3. **Começar escrevendo a seção *Responsabilidades*, descrevendo informalmente o propósito da operação.**
4. **Completar a seção *Pós-condições*, descrevendo declarativamente as mudanças de estado que ocorrem aos objetos do modelo conceitual:**

   - Criação e remoção de instância

   - Modificação de atributo

   - Formação e quebra de associações (erro mais comum!)

Como Fazer um Contrato

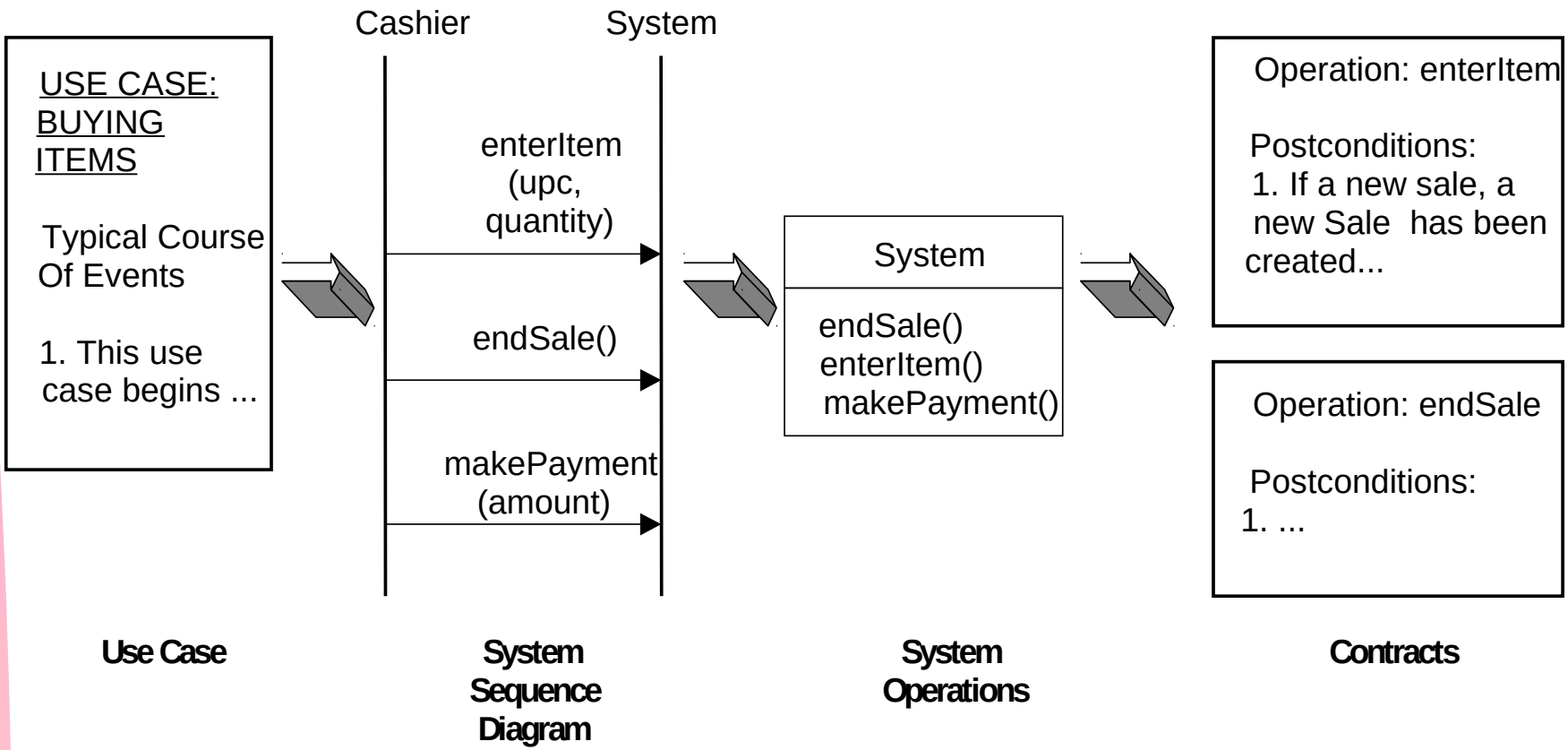
- **Regras úteis (cont.):**

  5. Completar a seção *Pré-condições*, descrevendo as pré-suposições sobre o estado do sistema no início da operação:

  - Coisas que devem ser testadas pelo sistema em algum ponto durante a execução da operação

  - Coisas que não são testadas, mas sobre as quais depende fortemente o sucesso da operação

Contratos e Outros Artefatos



Pós-condições: Palco e Cortina

- Pós-condições devem ser expressas no passado, enfatizando mudanças de estado já ocorridas

- Analogia: Palco e Cortina
  - Imagine os objetos do sistema no palco de um teatro
  - Antes da operação, fotografe o palco
  - Feche a cortina e execute a operação
  - Abra a cortina e fotografe o palco novamente
  - Compare as duas fotos, e então expresse como pós-condições as mudanças no estado do palco

Contratos para o Sistema POST

- **Operação *encerrarVenda*:**

Contrato:	encerrarVenda ()
Responsabilidades:	Registra o fim da entrada de itens de venda, e mostra o total da venda.
Tipo:	Sistema
Referencia:	Funções: R1.2 Casos de uso: Comprar Itens
Notas:	
Exceções:	Se não há venda corrente, indicar erro.
Saída:	
Pré-condições:	UPC é conhecido do sistema
Pós-condições:	

  - *Venda.Completada* é definido para *true* (modificação de atributo).

Note a necessidade de adicionar o atributo lógico *Completada* ao conceito *Venda*!

Contratos para o Sistema POST

- **Operação *fazerPagamento*:**

Contrato:	fazerPagamento (quantia: Número ou Quantidade)
Responsabilidades:	Registra o pagamento, calcula troco, e imprime recibo.
Tipo:	Sistema
Referencia:	Funções: R1.2
	Casos de uso: Comprar Itens
Notas:	
Exceções:	Se a venda não completada, indicar erro.
	Se quantia menor que total, indicar erro.
Saída:	
Pré-condições:	
Pós-condições:	

  - Um *Pagamento* foi criado (criação de instância).
  - *Pagamento.quantia* foi definido para *quantia* (modificação de atributo).

Contratos para o Sistema POST

- **Operação *fazerPagamento*:**

Pós-condições (cont.):

- Um *Pagamento* foi criado (criação de instância).
- O *Pagamento* foi associado com a *Venda* (formação de associação).
- A *Venda* foi associada com a *Loja*, para adicioná-la ao *log* de vendas completadas (formação de associação).

Contratos para o Sistema POST

Operação Inicializar:

Contrato: Inicializar ()

Responsabilidades: Inicializa o sistema.

Tipo: Sistema

Referencia:

Notas:

Exceções:

Saída:

Pré-condições:

Pós-condições:

- *Loja*, *POST*, *Catálogo-Produto*, e *Especificação-Produto* foram criados (criação de instância).
- POST foi associado com *Loja*, e *Catálogo-Produto* com *Especificação-Produto*, *POST*, e *Loja* (formação de associação).