Análise de Software
- Orientada a Objetos -

Prof. Ingrid Nunes

INF01127 - Engenharia de Software N

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Prosoft

.inf
INSTITUTO
DE INFORMÁTICA
UFRGS

Relembrando

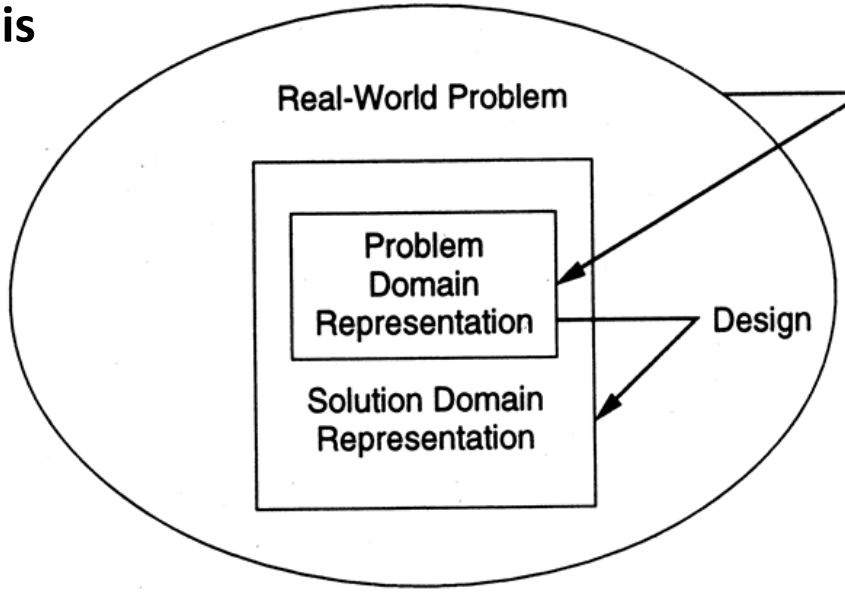
- Atividades genéricas em todos os processos de desenvolvimento de software
  - **Especificação**
    - O que o sistema deve fazer e suas restrições de desenvolvimento
      - Requisitos
      - Identificação, expressão, verificação
  - **Desenvolvimento**
    - Produção do sistema de software que atenda aos requisitos identificados
      - **Análise**
      - Projeto
      - Implementação
  - **Validação & Verificação**
    - Certificação de que o software está correto e é o que o cliente deseja

Análise vs. Projeto

- Fronteira mal definida !!!!!

mais orientado à análise — **mais orientado a projeto**

- **Investigação mais detalhada do problema e esboço de solução**
- **Requisitos funcionais**
- **Investigação do domínio**
- **Ponto de partida do projeto**
- **Solução conceitual**
- **O que?**



- **Desenvolvimento de solução lógica**
- **Requisitos funcionais e não funcionais**
- **Ambiente, tecnologia**
- **Solução lógica**
- **A um passo da implementação**
- **Como?**

Análise de Software

- Um esboço inicial do sistema
  - **Na prática, um modelo conceitual não é correto ou incorreto**
    - **Ele é mais ou menos útil**
- Orientado a **domínio**
- Abstração do projeto
  - Simplificação
  - Independente de plataforma ou tecnologia
  - Compreensão aprofundada dos requisitos
- **Temporário**
  - Trabalho de manter o modelo de projeto consistente com o modelo de análise deve ser cuidadosamente avaliado
  - Benefício de ter uma visão de alto nível que abstrai os principais detalhes de um sistema

Análise Orientada a Objetos

- Sistemas **procedurais** tradicionais
  - **Separam dados e procedimentos**
  - Modelam estes **separadamente**
- Orientação a **objetos**
  - Vê **dados e funções em conjunto**
  - Base é a **abstração** de **dados**
- Objetivo da análise e projeto OO
  - Definir as classes no sistema a serem construídas e seus relacionamentos

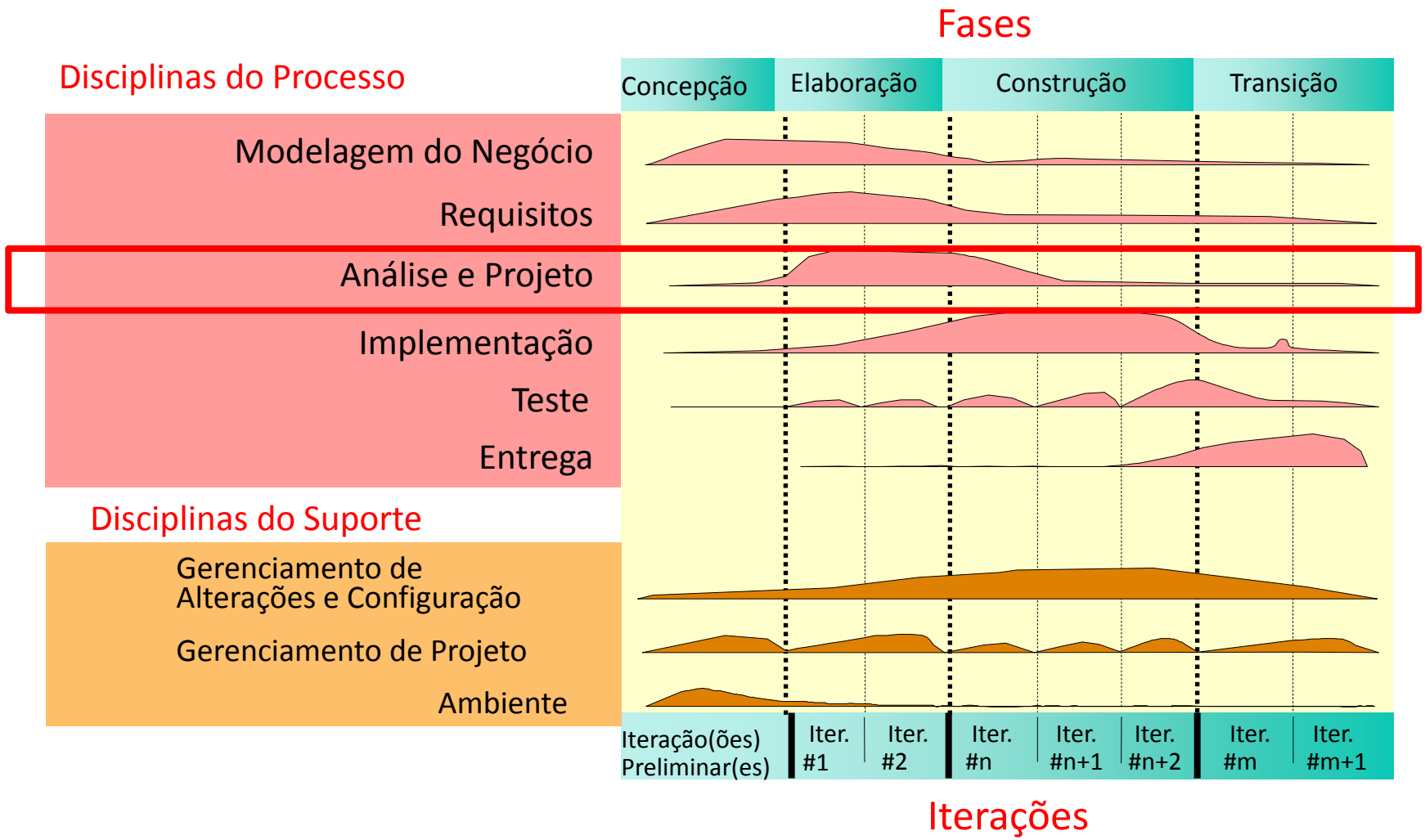
Análise Orientada a Objetos

- Técnica Semi-formal especificação
- Diferentes métodos
  - Booch
  - OMT
  - Objectory
  - Shlaer-Mellor
  - Coad-Yourdon
- Todas essencialmente equivalentes
- Hoje
  - Representada usando UML

Análise de Software

ANÁLISE DE SOFTWARE NO RUP

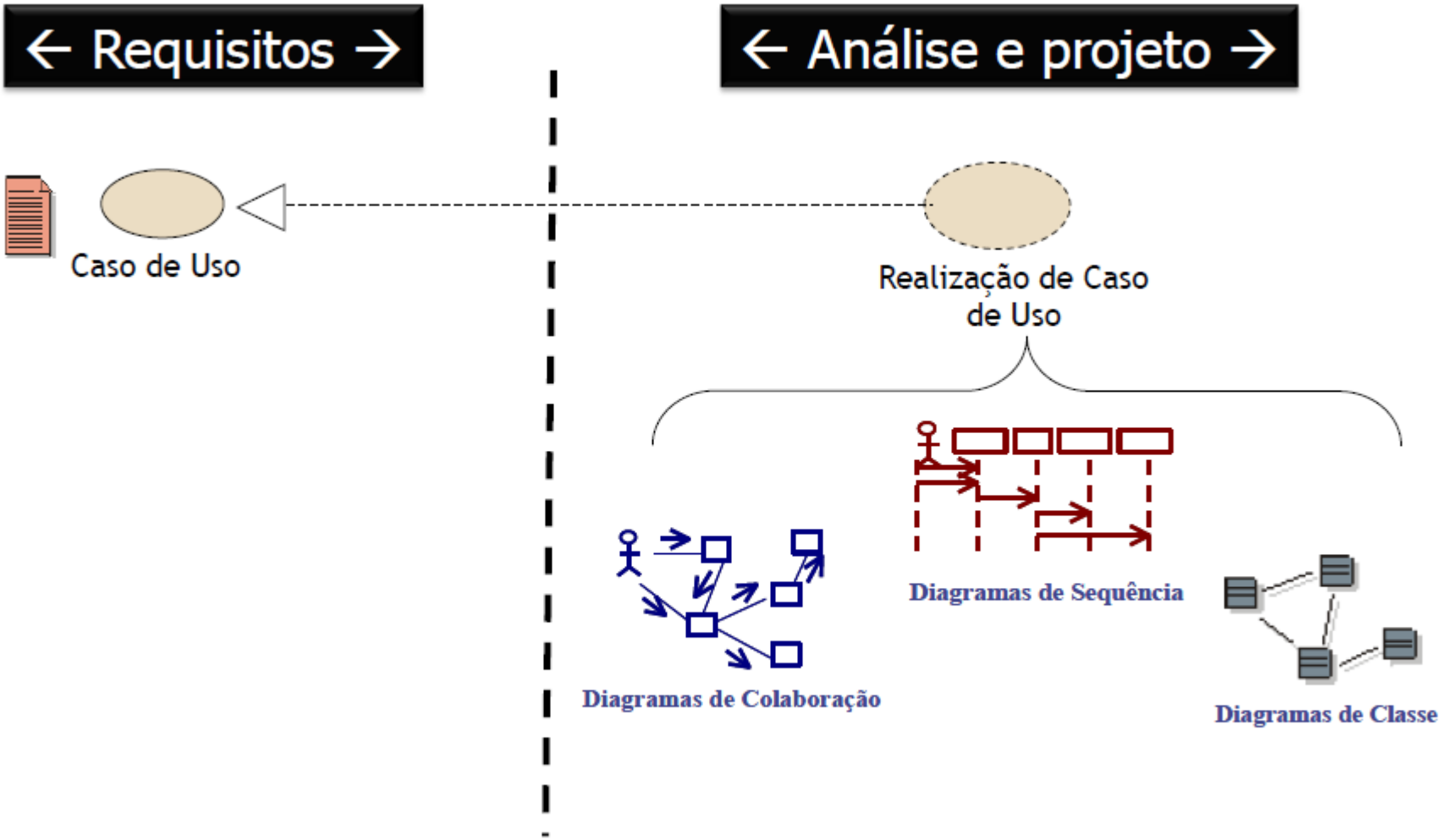
Processo Unificado: Relembrando



RUP: Análise e Projeto

- Objetivos básicos
  - Criar um **projeto** a partir dos **requisitos** identificados
  - Derivar uma **arquitetura** para o sistema
  - Adaptar o projeto para as **limitações** do **ambiente** de execução
- Centrado em **Realização** de **Casos** de **Uso**
  - Iterações
  - Rastreabilidade
- Pode ser dividido em dois modelos
  - Modelo de Análise (transitório)
  - Modelo de Projeto (permanente)

Realização de Casos de Uso



Modelo de Análise

- Objetivo
  - Representação dos **requisitos funcionais** do sistema
  - Representação de **conceitos do domínio** de problema
- RUP é centrado na **realização de casos de uso**
  - Incentiva a busca das classes envolvidas na realização de cada caso de uso
- Realização de Casos de Uso – Análise
  - Diagrama de Classes (**de análise**)
  - Diagrama de Interação
    - Colaboração entre classes decorrente de um evento gerado por um ator
    - Delineia fluxo de eventos no sistema
  - Pacotes de serviços

Modelo de Análise vs. de Casos de Uso

Modelo de Casos de uso

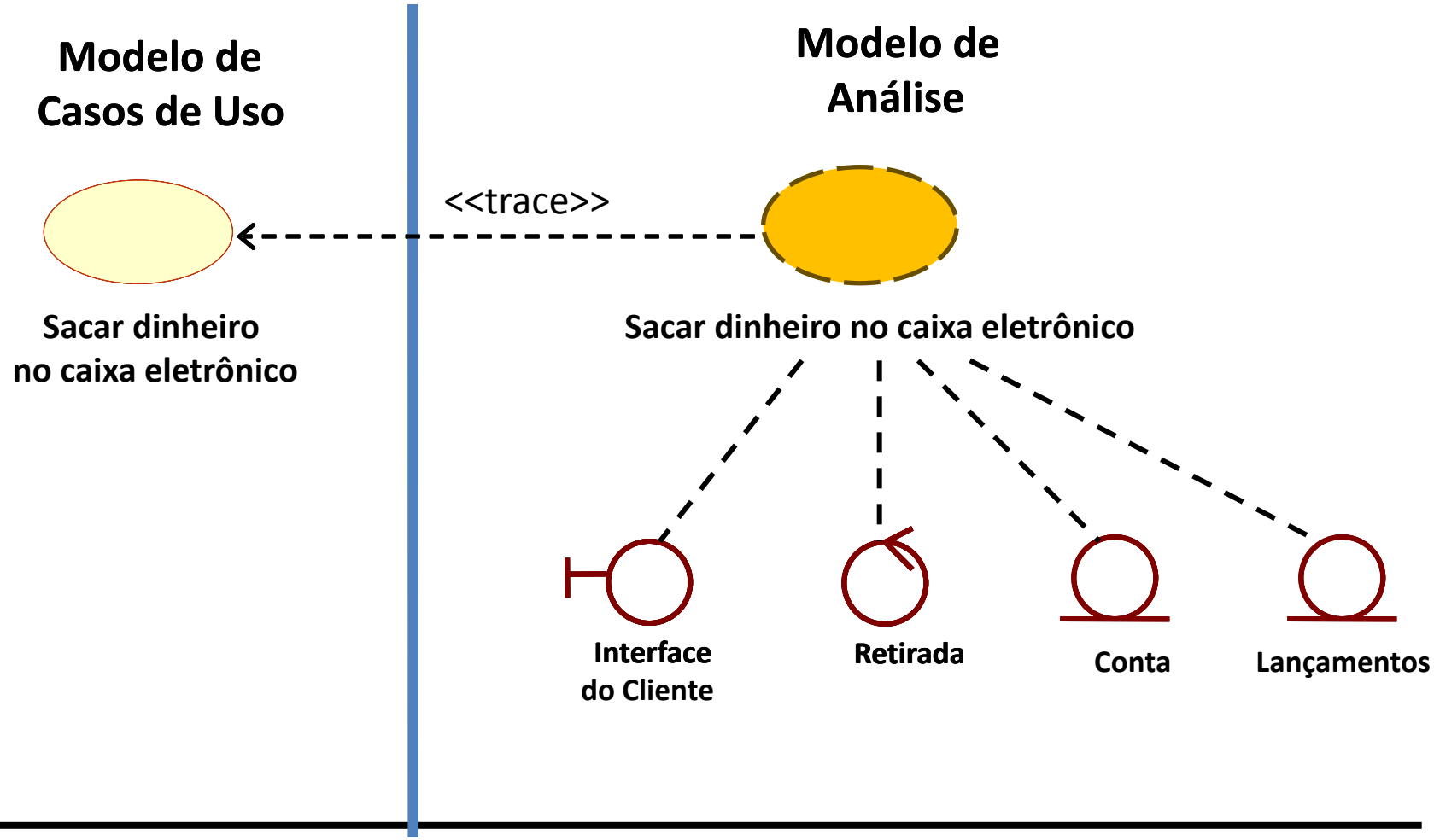
- Descrito na linguagem do cliente
- Visão externa do sistema
- Estruturada por casos de uso que estruturam a visão externa
- estabelece contrato entre cliente e desenvolvedores sobre o quê o sistema deve fazer
- Pode conter redundâncias, inconsistências, etc. entre requisitos
- Captura a funcionalidade do sistema
- Define os casos de uso que são endereçados no modelo de análise

Modelo de Análise

- Descrito usando a linguagem do desenvolvedor
- Visão interna do sistema
- Estruturado por classes estereotipadas e pacotes que estruturam a visão interna
- Auxilia desenvolvedores a compreender a forma inicial do sistema
- Serve para reduzir inconsistências ou ambiguidades
- Esboça como concretizar a funcionalidade no sistema
- Define realizações de casos de uso

CONSTRUÇÃO

Realização de um Caso de Uso



Realização de Caso de Uso

- Descreve como o caso de uso é realizado
  - Associa caso de uso com classes e outros elementos de projeto
- Utiliza diagramas
  - Classes (de análise)
    - Para organizar classes em um nível de detalhe relevante para a análise
  - Interação
    - Para distribuir comportamento
- Por que investir em classes de análise?
  - Modelo de classes transitório
    - **Ponto de partida**
      - Serão convertidas para **classes de projeto**
    - **Transição**
      - Diminuem a distância entre os requisitos e o projeto
  - Favorecem a **rastreabilidade** e a **justificativa** de modelagem

Modelo de Análise

- Fonte de **Referência**
  - **Casos** de **uso expandido**
- **Classes** de Análise
  - Representam o conceito mais abstrato dos elementos do sistema
    - Definem **atributos**, porém **sem** o compromisso com **detalhamento**
    - Raramente definem comportamento (operações e assinaturas)
      - "responsabilidades"
  - Estabelecem suas **relações** com outras classes
  - "Isolamento de preocupações" (***separation of concerns***) padrão
    - Classes com papéis de abstrair aspectos específicos

Modelo de Análise: Passo

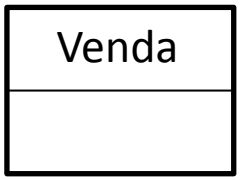
1. **Identificação dos objetos**
   - Usando conceitos, cartões CRC, estereótipos, etc.
2. **Organizar os objetos**
   - Classificar os objetos identificados
   - Objetos de semelhantes podem ser definidos mais tarde na mesma classe
3. **Identificar as relações entre objetos**
   - Isso ajuda a determinar as entradas e saídas de um objeto
4. *Definindo operações dos objetos*
   - *Forma de processamento de dados dentro de um objeto*
5. *Definindo objetos internamente*
   - *Informações contidas nos objetos*

Diretrizes para Modelos de Análise

- Esboço de um diagrama de classes
- Manter o modelo em uma ferramenta
  - Se o modelo precisar ser mantido e atualizado, utilizar ferramenta case
- Pense como um cartógrafo (pág. 171, Larman)
  - Use os nomes existentes no território
  - Excluir características irrelevantes ou fora do escopo
  - Não incluir coisas que não estão lá
- Como modelar o mundo irreal
  - Domínios com pouca analogia com domínios naturais ou de negócios
    - Ex.: software para telecomunicações

Diretrizes para Modelos Conceituais

- Engano comum relativo a atributo vs. classes
  - Regra: se não pensamos em alguma classe conceitual X como um número ou texto no mundo real, X provavelmente é uma classe conceitual, não um atributo



Loja ≠ número ou texto

- Quando modelar com classes descritivas
  - Houver a necessidade de uma descrição sobre um item ou serviço
  - Ela reduzir informação redundante ou duplicada
  - Ex.: classe DescriçãoDoProduto

Descobrindo Classes de Análise

- Estratégias
  - Relacione os conceitos candidatos do domínio do problema a pertencer a lista de categorias* de conceitos
    - Objetos físicos, transações, linhas de itens de transações, papéis desempenhados por pessoas, contêineres de coisas, eventos, etc.
  - Identificar os substantivos e frases que podem estar no lugar de um substantivo nas descrições do domínio do problema e considerá-los como candidatos a conceitos ou atributos para o modelo conceitual

* Para uma lista completa das categorias propostas por Larman www.dsc.ufcg.edu.br/~jacques/cursos/apoo/html/anal1

Refinando Classes de Análise

- Identificação dos **atributos**
  - Mas sem compromisso de detalhá-los
- Identificação de **associações**
  - Quais existem
  - Cardinalidades e restrições
- Identificação de **generalizações**
- Identificação das **responsabilidades** genéricas, a partir dos papéis desempenhados nos casos de uso
  - Diagramas de colaboração

REPRESENTAÇÃO

Análise

- Três formas de fazer e representar modelos de análise orientada a objetos
  - **Modelo Conceitual** (Larman)
    - Produz um diagrama de classes “light”
  - **Cartões CRC** (Beck, Cunningham)
    - Cartões de índice e atuação de papéis
  - **Modelo de análise com estereótipos** (Jacobson)
    - Fronteiras, entidades, controle

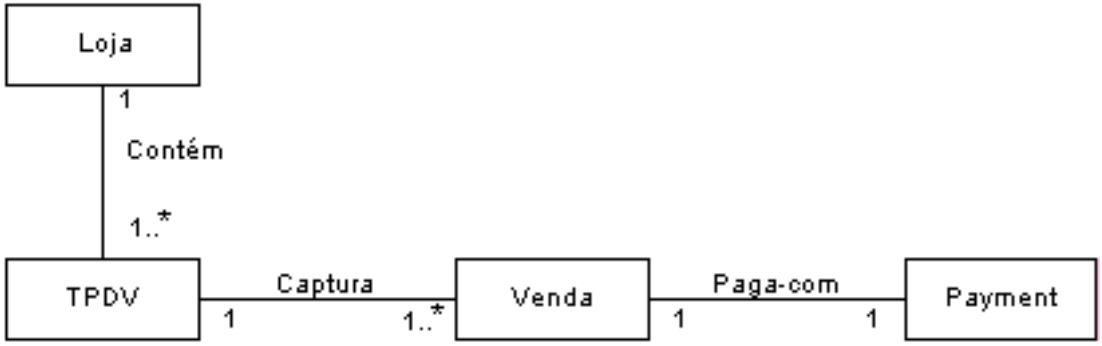
Modelo Conceitual

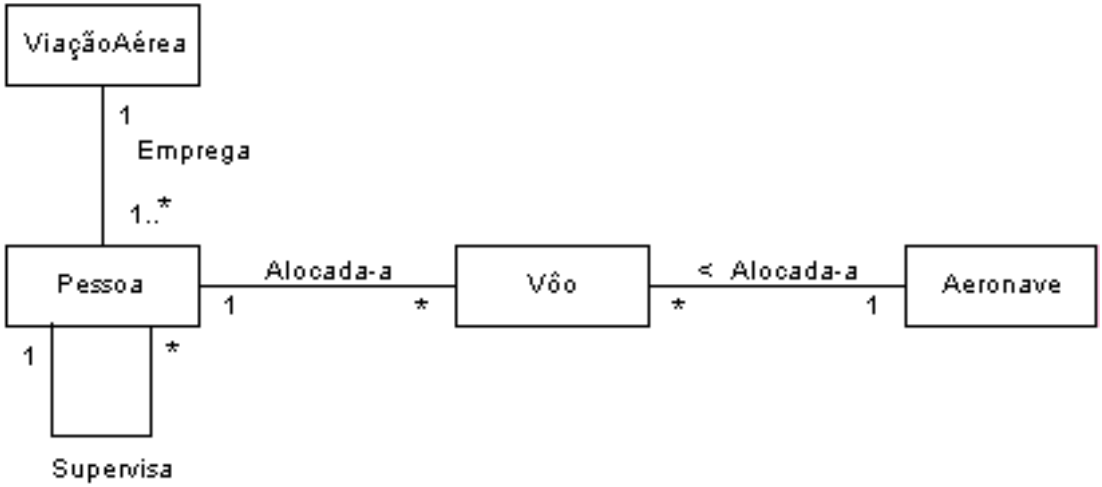
Um modelo de domínio é uma representação visual de classes conceituais, ou objetos do mundo real, em um domínio

Fowler, 1196

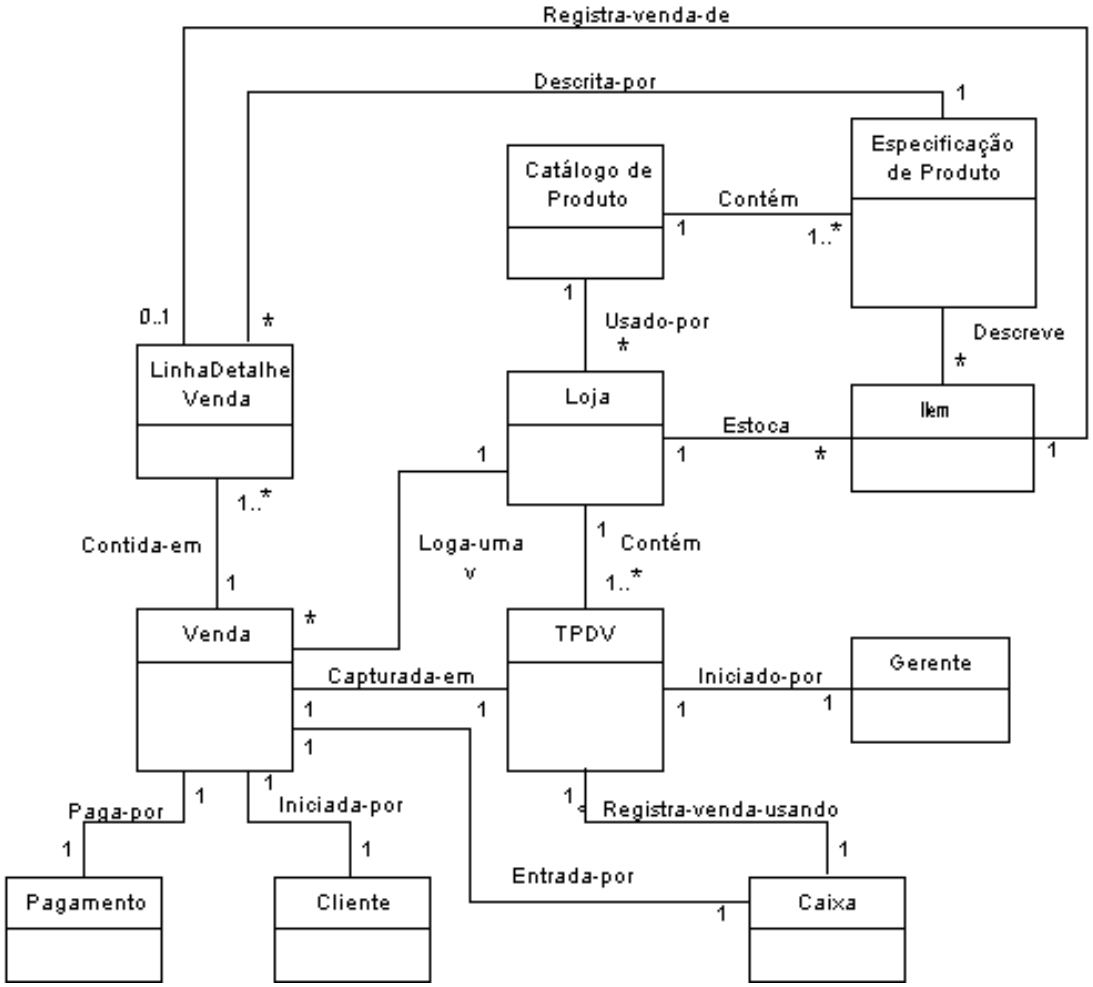
- Modelo conceitual
  - Ilustra os **conceitos significativos** (para os modeladores) em um **domínio** do problema
    - Representação do domínio, e **não** dos **componentes** de software
    - **Decomposição** do espaço do problema em unidades compreensíveis (**conceitos**)
    - Compreensão do **vocabulário** do domínio, comunicando às partes interessadas os termos relevantes e como eles estão relacionados

Modelo Conceitual: Exemplos





Modelo Conceitual: Exemplos



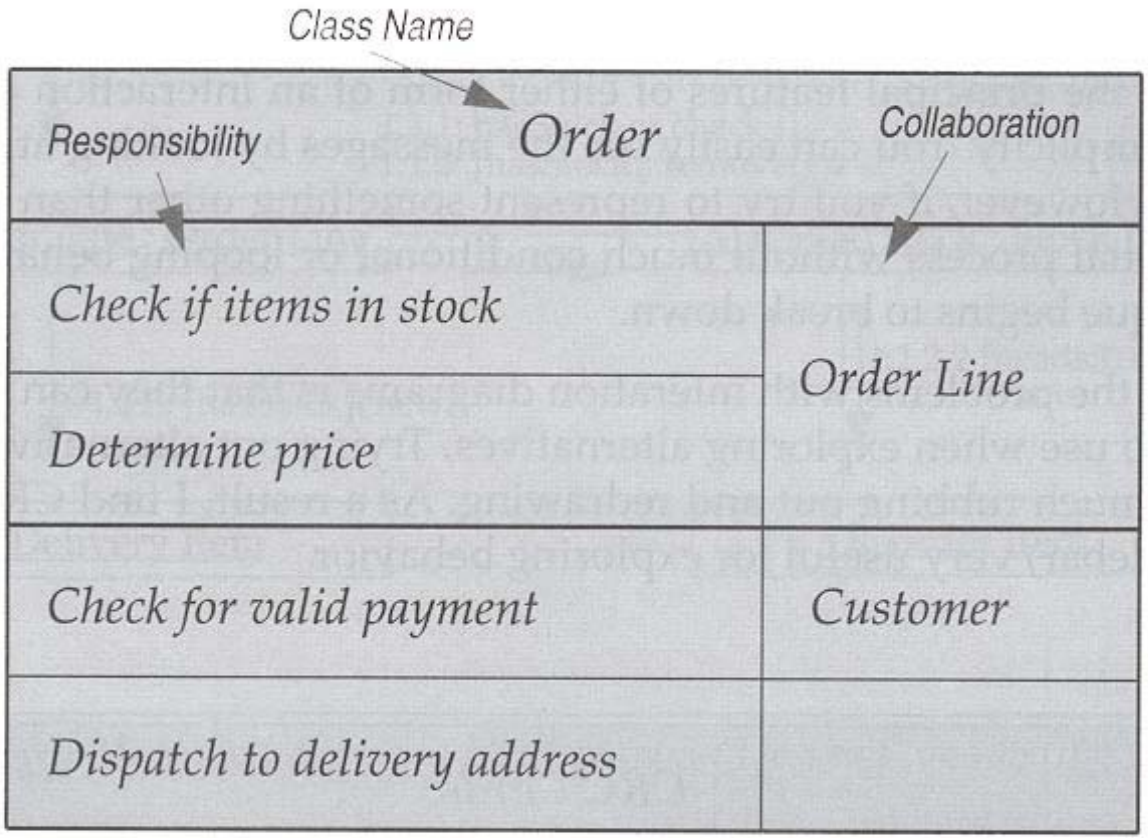
Cartões CRC

- Cartões CRC (**C**lass-**R**esponsibility-**C**ollaborator)

Nome	Responsabilidades
Colaboradores	

  - Um por classe, mostrando
    - Suas **responsabilidades**
    - Com a qual(is) outra(s) classe(s) que deve **colaborar** para cumprir cada responsabilidade
  - Breve **descrição** da classe na parte de trás do cartão
- Úteis na detecção de responsabilidadesde objetos
  - Desenvolvidos por Kent Beck eWard Cunningham

Cartões CRC



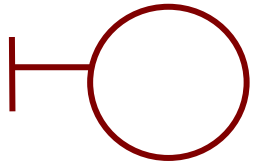
Class-Responsibility-Collaboration (CRC) Card

Modelo de Análise com Estereótipos

- Estratégias RUP: identificar
  - **Classes entidade**
    - A partir do estudo **da descrição** do caso de uso e do modelo do domínio (se existir)
    - Considerando qual a informação envolvida e manipulada na realização de um caso de uso (classes vs. atributos)
  - Uma **classe fronteira**
    - Central para cada ator (interface primária)
  - Uma **classe controle**
    - Responsável pelo controle e coordenação do caso de uso

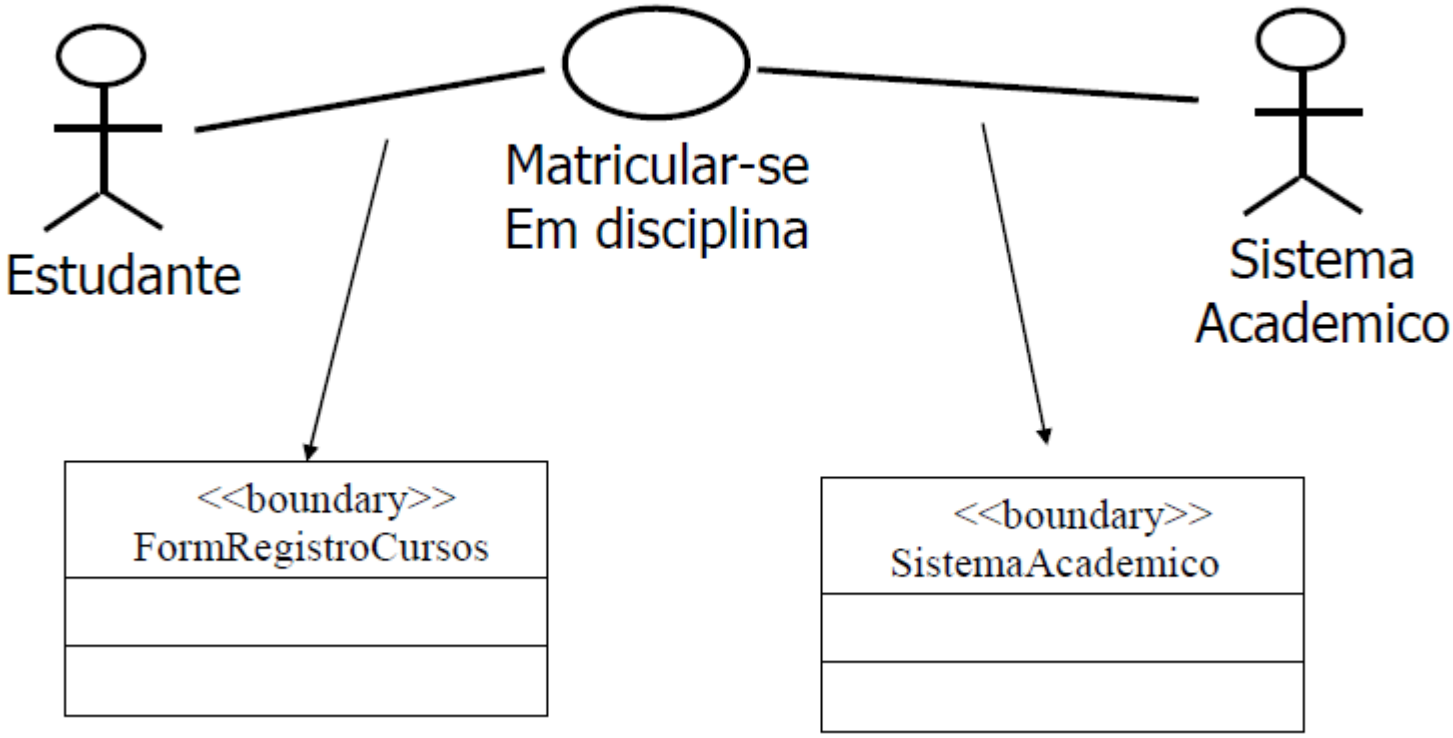
Modelo de Análise com Estereótipos

- Classe **Fronteira** (boundary)



  - **Abstrai** uma **interação** entre sistema e ator
    - Interface gráfica, formulários Web
    - Importante quando é preciso definir uma interface para o sistema
    - Desnecessária em sistemas muito simples
  - Cada classe de fronteira deve estar relacionada a pelo menos um ator e vice-versa
    - Mais detalhamentos ou outras formas de agrupamento são possíveis, mas a utilidade é questionável
  - Detalhamento não é relevante na análise de domínio
    - Projeto de interface com usuário detalha a interação
    - Projeto pode levar em conta especificidades de tecnologias de dispositivos ou de arquiteturas de integração
  - Importância
    - **Saber** quem ou o quê **interage** com o **sistema**

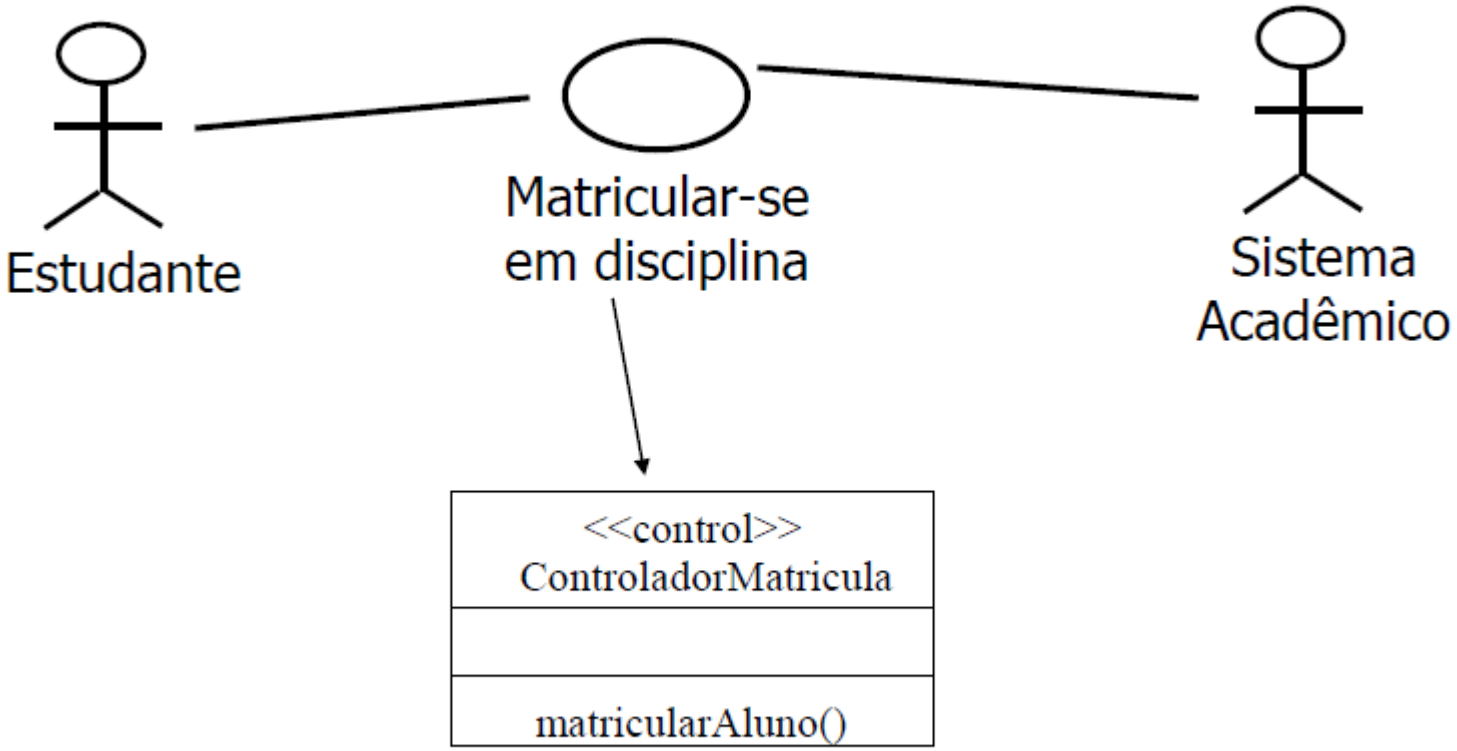
Modelo de Análise com Estereótipos



Modelo de Análise com Estereótipos

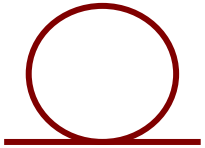
- Classe de **Controle** (control)
  - **Abstraem** controle relacionado a um caso de uso específico
    - Representam coordenação, sequenciamento, transações e controle de outros objetos
    - Representam processamento complexo (tal como cálculos envolvidos na lógica de um processo de negócio) que não seja apropriado a uma classe entidade em particular
  - Importância: refletir sobre
    - Quais **entidades** deve conhecer e manipular
    - Quais **regras** de negócio e restrições governam o controle
    - Qual **interface ativa** o processamento e recebe a resposta
  - Tipicamente, um único controle por caso de uso
    - É possível fazer mais detalhamento, mas utilidade é questionável

Modelo de Análise com Estereótipos

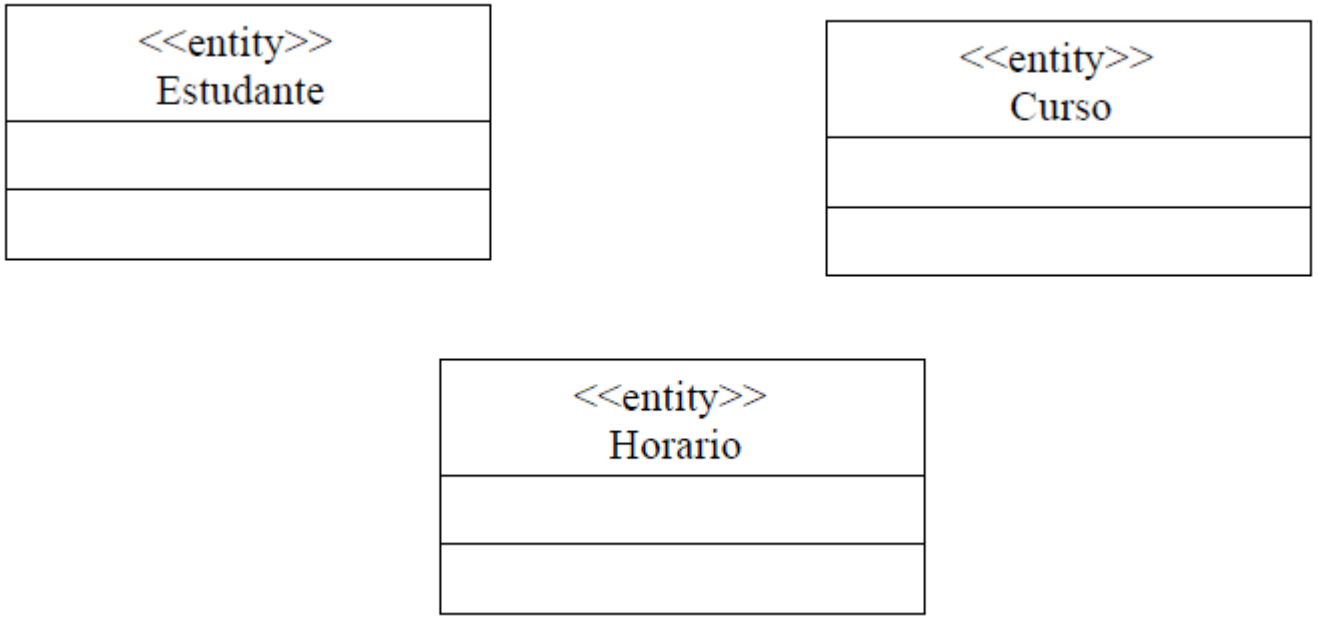


Modelo de Análise com Estereótipos

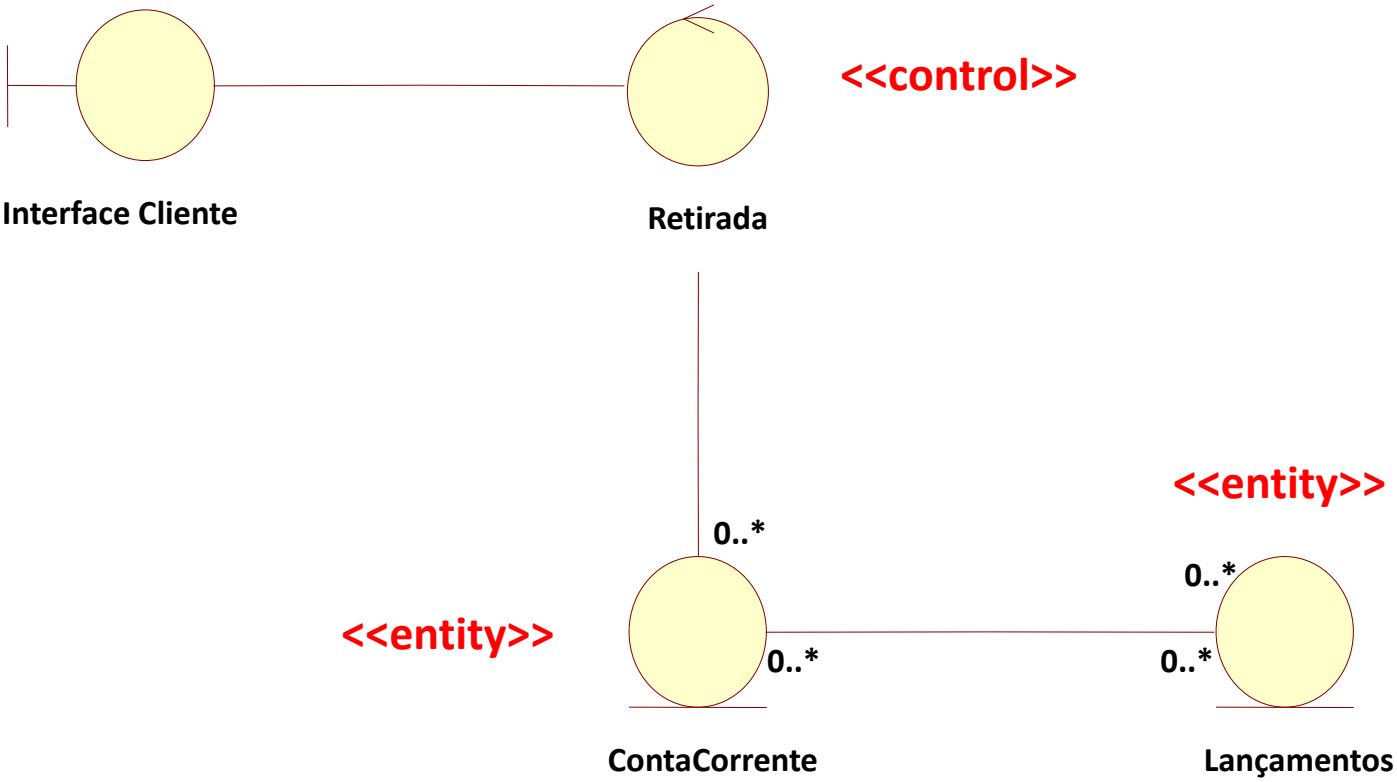
- Classe **Entidade** (entity)
  - De **importância inquestionável**
    - Maior foco de atenção no modelo de análise
  - Modela **informações** e **comportamentos** associados a um conceito ou fenômeno do sistema
  - Utilizada para **abstrair informações** relacionadas que têm longa vida no sistema
    - Frequentemente (mas não necessariamente) são persistentes
    - Pode ter um comportamento complexo relacionado à informação que representa
  - Frequentemente mostra uma estrutura lógica de dados e contribui à compreensão de qual informação o sistema depende



Modelo de Análise com Estereótipos

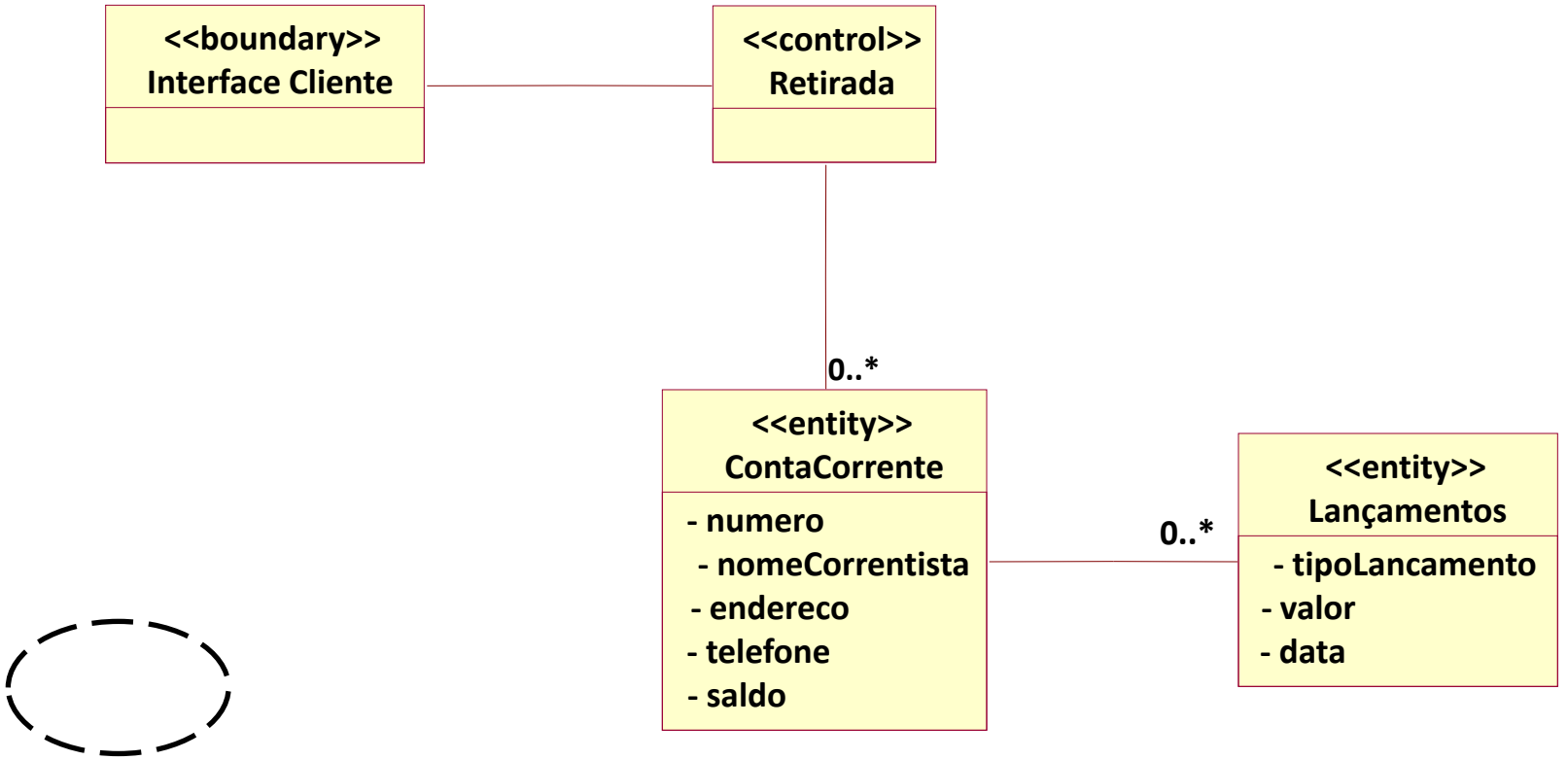


Modelo de Análise com Estereótipos: Exemplo



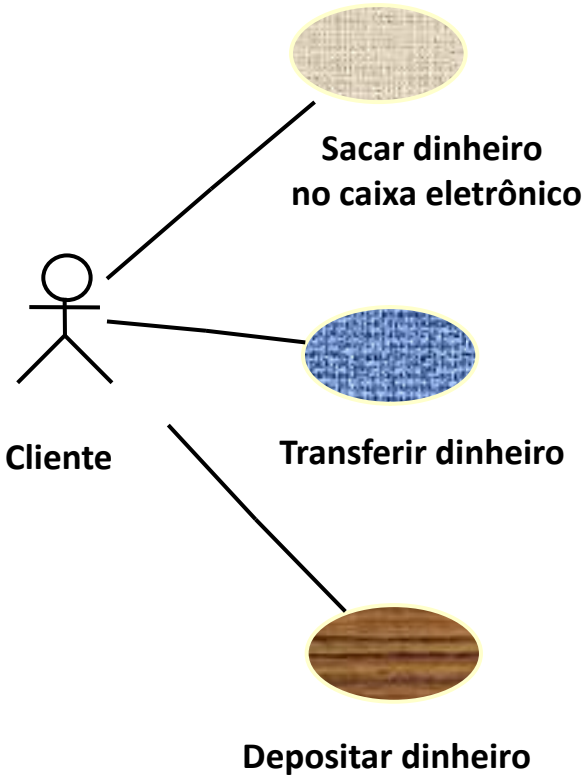
Sacar dinheiro no caixa eletrônico

Modelo de Análise com Estereótipos: Exemplo

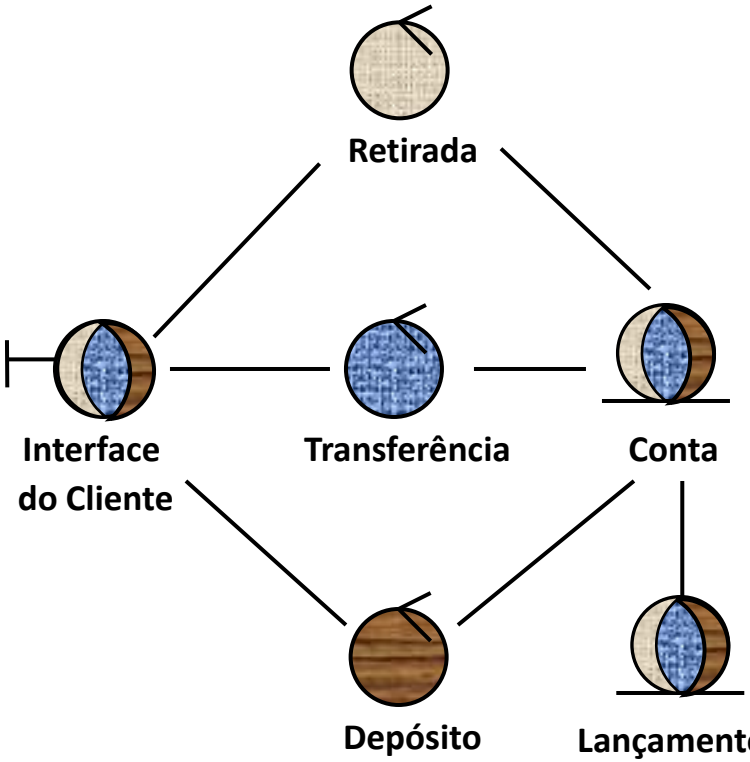


Modelo de Análise com Estereótipos: Exemplo

Modelo de Casos de Uso



Modelo de Análise



EXEMPLO

Exemplo: Caso de Uso Logar-se

Identificação: UC1

Caso de uso: Logar-se

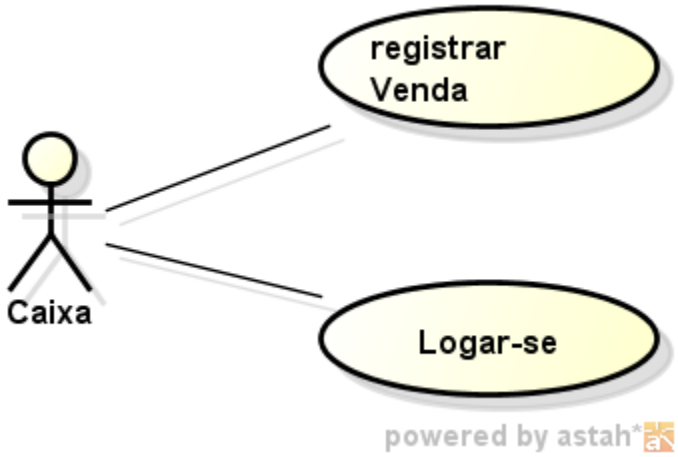
Atores: Atendente

Pré-Condições:

Pós-Condições: é registrada a operação de um terminal por um atendente identificado.

Sequência Típica de Eventos (Fluxo Básico):

1. Este caso de uso começa quando o Atendente fornece sua identificação (login e senha) ao terminal
2. O sistema habilita o registro de vendas naquele terminal, registrando data e hora do inicio da operação do terminal pelo Atendente.



Exemplo: Caso de Uso Comprar Itens

Identificação: UC2

Caso de uso: Registrar Venda

Atores: Atendente

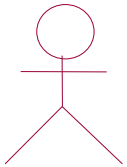
Pré-Condições: o Atendente está logado no terminal

Pós-Condições: a venda é registrada, o estoque do produtos vendidos é atualizado, o pagamento é registrado e o recibo do cliente é impresso.

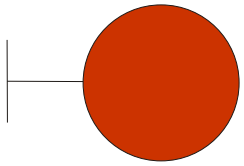
Sequência Típica de Eventos (Fluxo Básico):

1. O Atendente inicia uma nova venda
2. Para cada produto
    1. O Atendente entra com a identificação e quantidade de itens do produto
    2. O sistema registra a venda do produto e exibe no terminal sua descrição, preço e o valor total da venda de acordo com a quantidade de itens
3. O Atendente informa o término da venda
4. O sistema exibe o valor total da venda
5. O Atendente registra o pagamento recebido
6. O sistema registra a venda como completa, e imprime um recibo contendo dados da venda e do terminal onde esta foi efetuada , junto com que o operava.

Classes de Análise - Fronteira

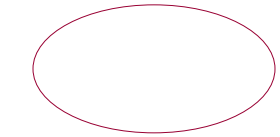


Atendente

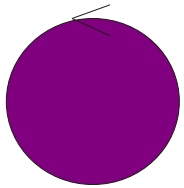


GUI Atendente

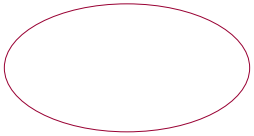
Classes de Análise - Controle



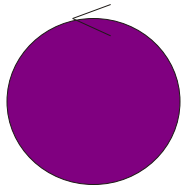
Registrar Vendas



registro Venda



Logar-se



logIn

Exemplo: Caso de Uso Registrar vendas

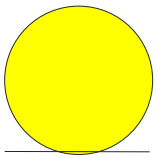
Identificação: UC2
Caso de uso: Registrar vendas
Atores: Atendente
Pré-Condições: o Atendente está logado no terminal

Pós-Condições: a venda é registrada, o estoque do produtos vendidos é atualizado, o pagamento é registrado e o recibo do cliente é impresso.

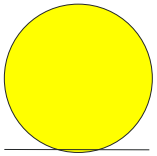
Sequência Típica de Eventos (Fluxo Básico):

1. O Atendente inicia uma nova venda
2. Para cada produto
    1. O Caixa entra com a identificação e quantidade de itens do produto
    2. O sistema registra a venda do produto e apresenta a sua descrição, preço e o valor total da venda de acordo com a quantidade de itens
3. O Atendente informa o término da venda
4. O sistema apresenta o valor total da venda
5. O Atendente registra o pagamento recebido
6. O sistema registra a venda como completa, e imprime um recibo contendo dados da venda e do terminal onde esta foi efetuada, junto com que o operava.

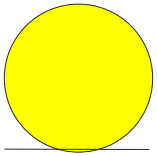
Classes de Análise - Entidades



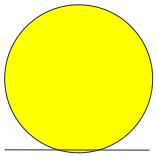
Terminal



Venda

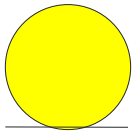


Produto



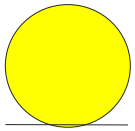
Atendente

Classes de Análise - Refinamento



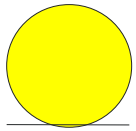
Terminal

identificação



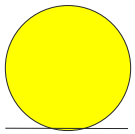
Venda

data
numero
vtotal



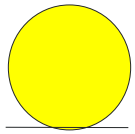
Produto

identificação
descrição
preço
quantidade



Atendente

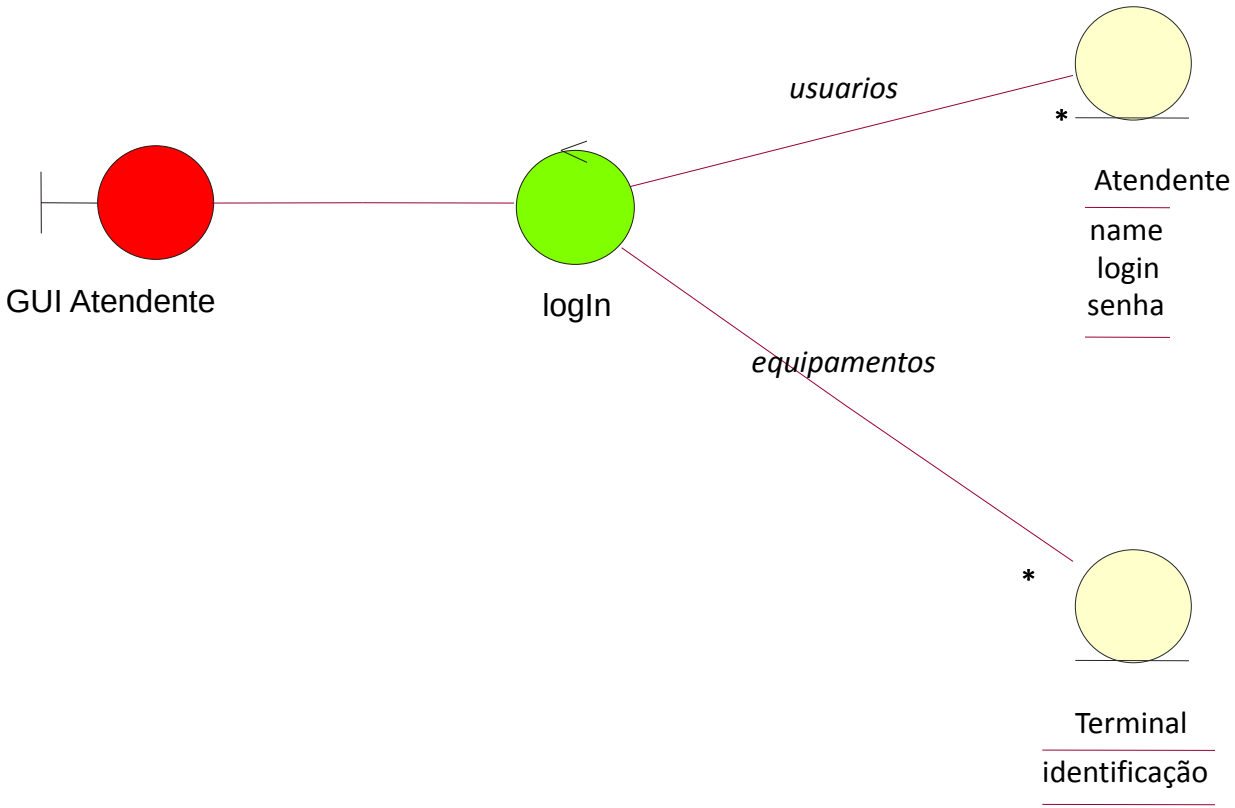
name
login
senha

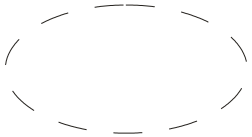


itemVenda

quantidade

Classes de Análise - Refinamento





Logar-se

Relação de conhecimento entre controle e entidade é muito importante

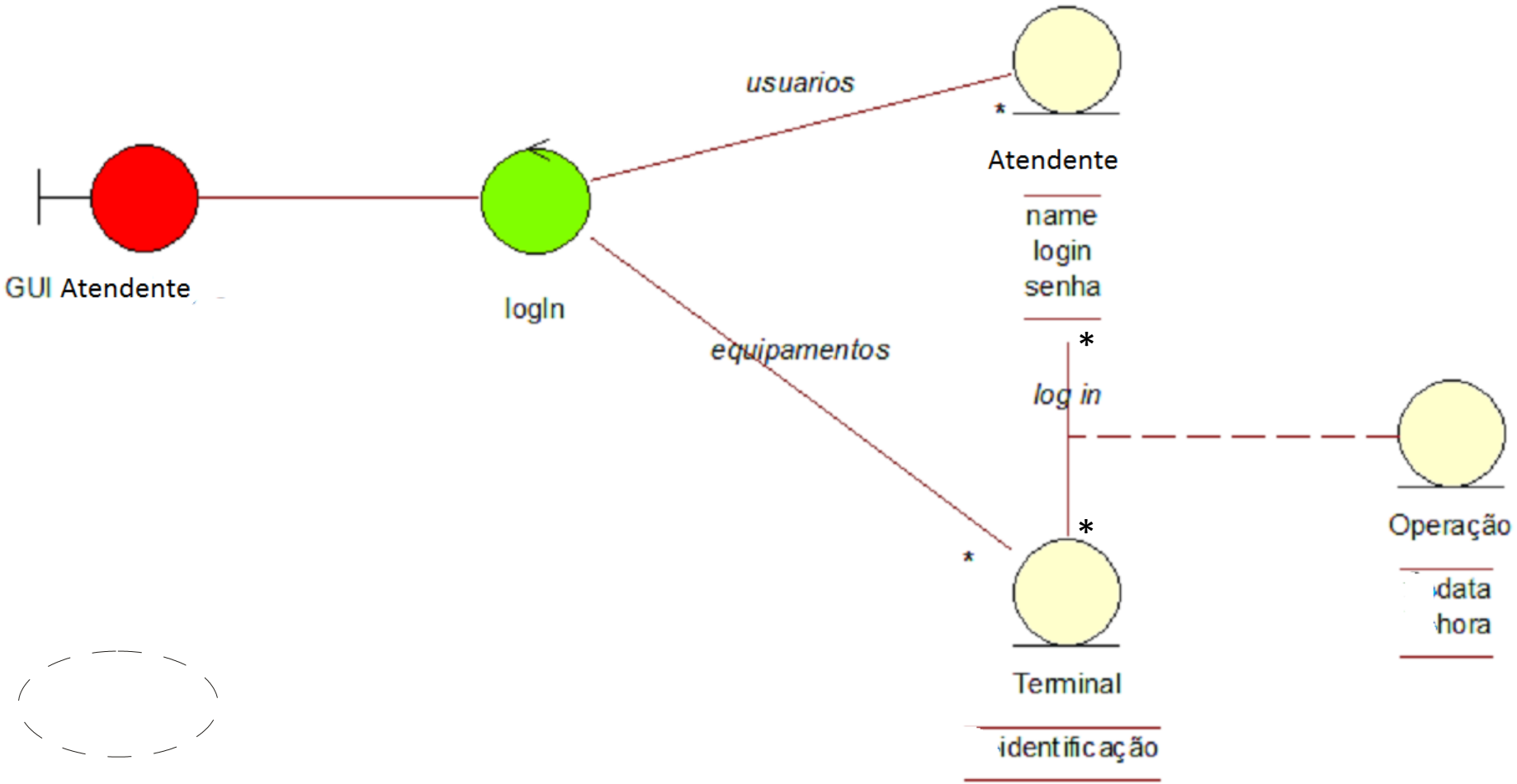
Exemplo: Caso de Uso Logar-se

Identificação: UC2
Caso de uso: Logar-se
Atores: Atendente
Pré-Condições:
Pós-Condições: é registrada a operação de um terminal por um Atendente identificado.

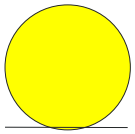
Seqüência Típica de Eventos (Fluxo Básico):

1. Este caso de uso começa quando o Atendente fornece sua identificação (login e senha) ao terminal
2. O sistema habilita o registro de vendas naquele terminal, registrando data e hora do inicio da operação do terminal pelo Atendente.

Classes de Análise - Refinamento

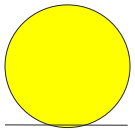


Classes de Análise - Refinamento



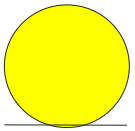
Terminal

identificação



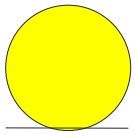
Venda

data
numero
vtotal



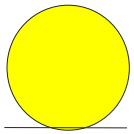
Produto

identificação
descrição
preço
quantidade



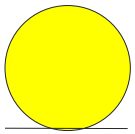
Operação

data
hora



itemVenda

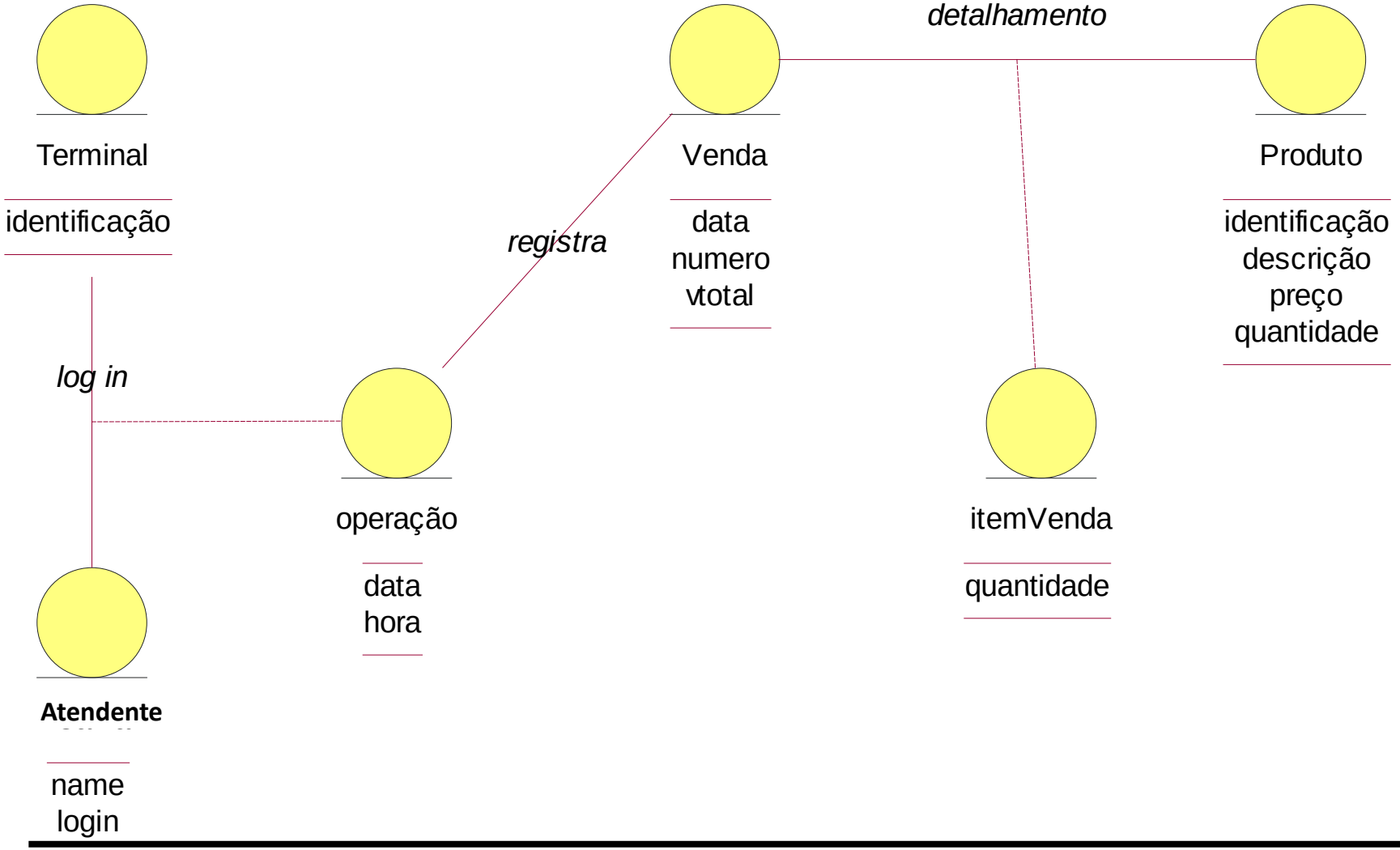
quantidade



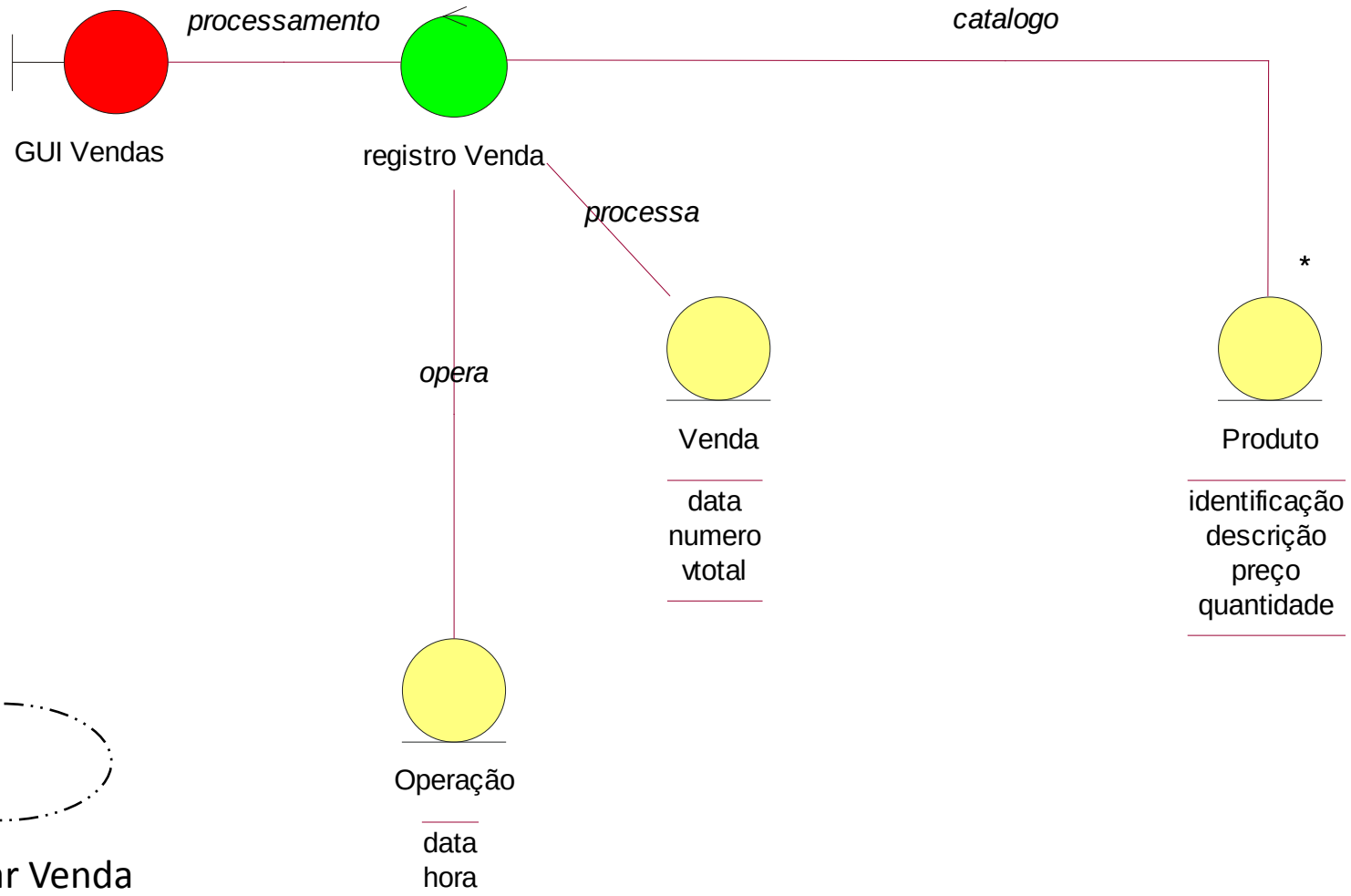
Atendente

name
login
senha

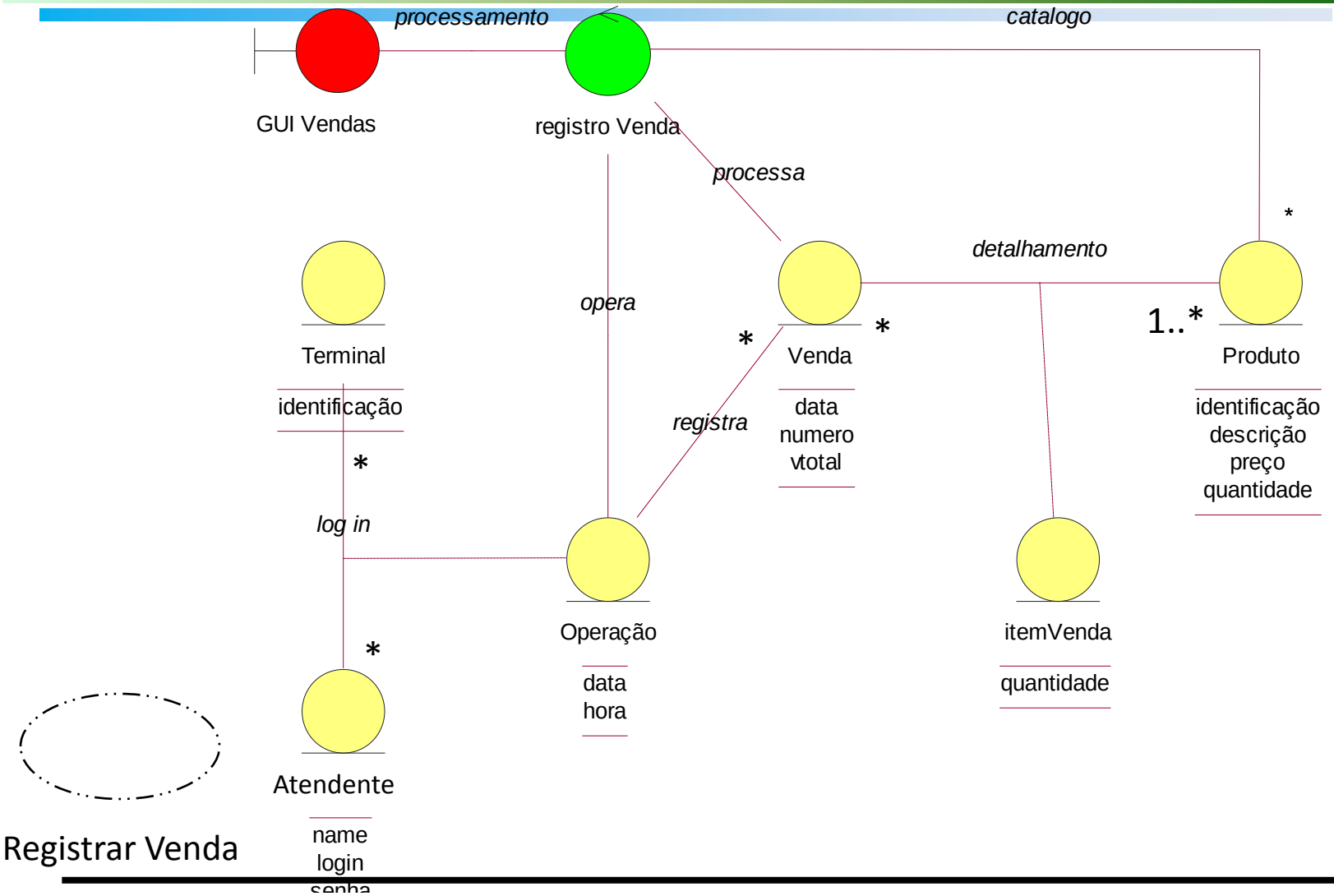
Classes de Análise - Refinamento



Classes de Análise - Refinamento



Classes de Análise - Refinamento

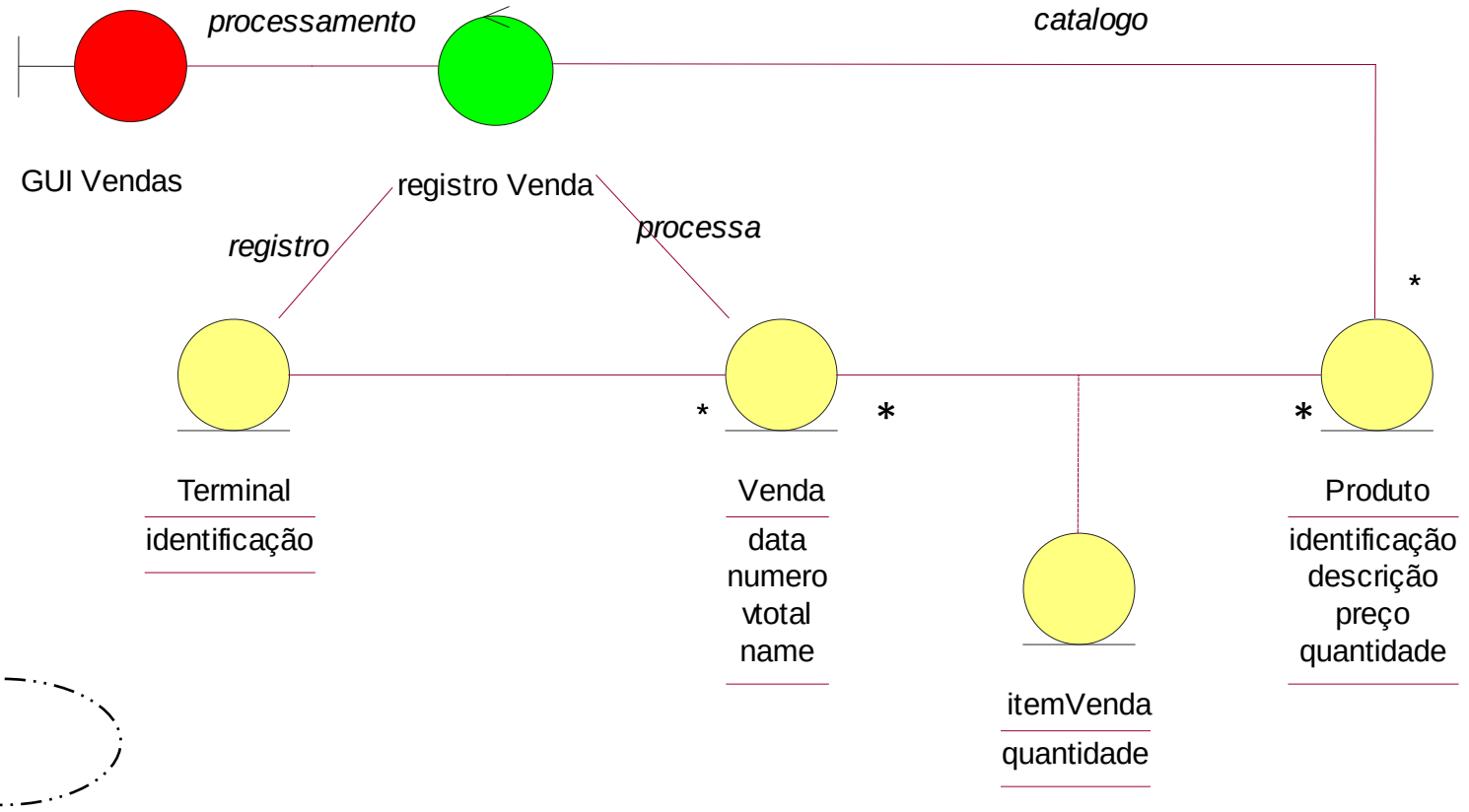


Registrar Venda

Realização de Caso de Uso

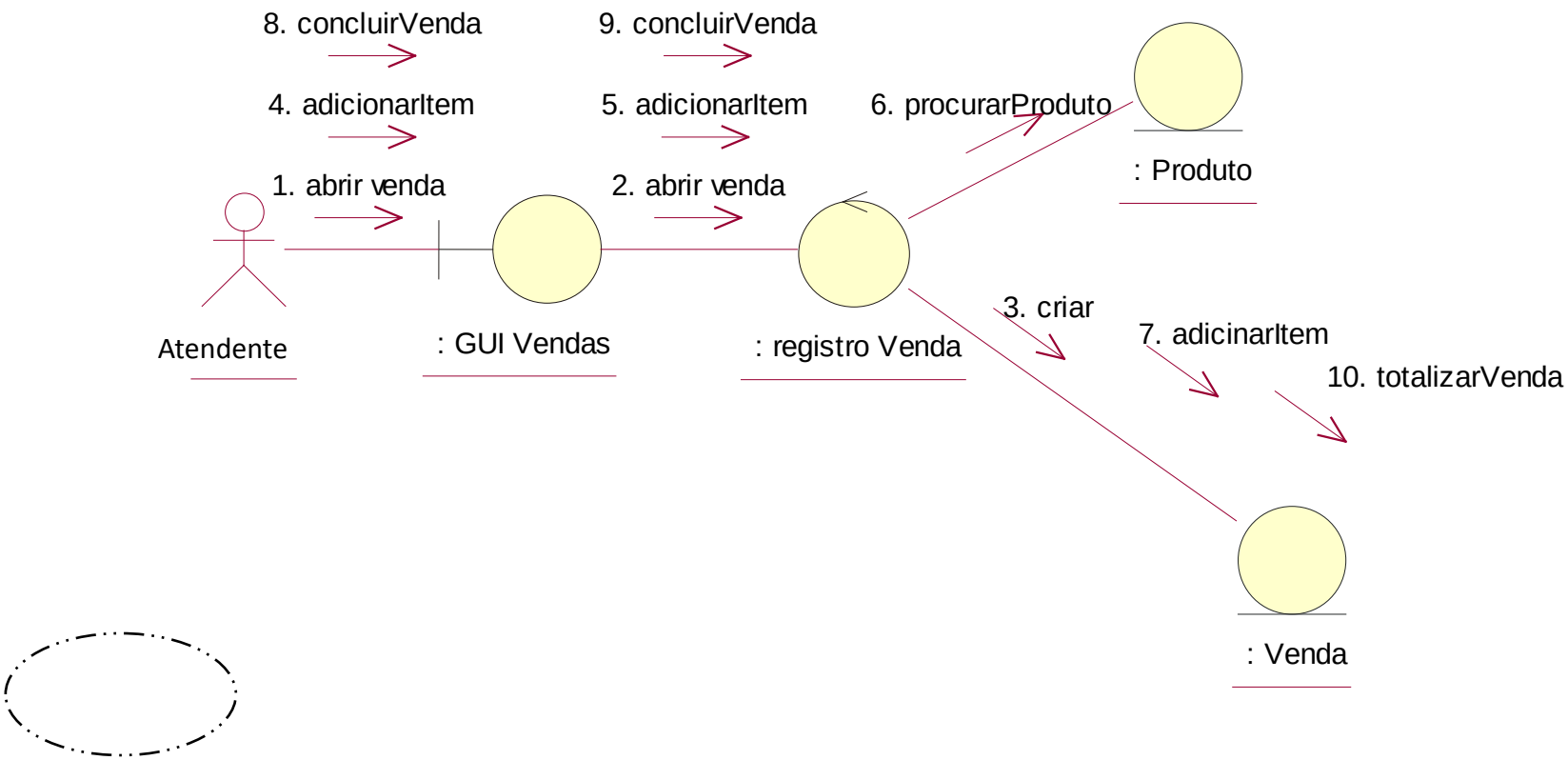
- Diagrama de Classes
  - Estático
    - Vocabulário
    - Relacionamentos e regras sobre a informação
    - Esquema conceitual da base de dados
- Diagramas de colaboração
  - Dinâmico
    - Colaboração em nível de responsabilidades genéricas
- Diagramas são complementares

Realização: Diagrama de Classes



Registrar Venda

Realização: Diagrama de Colaboração



RegistrarVendas

Modelagem: Nunca se Esqueçam

- Objetivos
  - **ENTENDER**
  - **COMUNICAR**
- Modelos/diagramas não são necessariamente
  - Documentos
  - Permanentes
  - Organização/processo definem
    - Processo unificado é centrado na comunicação via modelos documentados

Modelo de Análise: Considerações

- **Temporário**: Um modelo conceitual preliminar de como o sistema funcionará
  - Evolui rapidamente na fase de projeto
  - Permite estudar implicações de diferentes alternativas de projeto
  - Permite entender com mais detalhes os requisitos do sistema e o domínio
- Lembre-se
  - **Modelo de análise tipicamente não é certo ou errado**
    - **Ele é mais ou menos útil**
- **Classes** de análise **evoluem** durante projeto
  - "Proto-classes" representando agrupamento de comportamento
    - Dão origem a novas classes (ou conjuntos de classes), ou mesmo a subsistemas
  - Frequentemente representam colaborações de múltiplos objetos
  - Permitem explorar distribuições alternativas de responsabilidades para atingir uma boa separação de preocupações

Modelo de Análise: Considerações

- Quando um modelo de análise tem maior utilidade?
  - Ainda **não** se **conhece** o **domínio**
    - Inexperiência no desenvolvimento de aplicações no domínio
  - Aplicação é **complexa**
    - Processo de investigação do domínio é extenso
  - Sistema deve ser **projetado** para **vários ambientes** alvo, com arquiteturas de projeto distintas
    - Modelo de Análise é uma abstração do Modelo de Projeto
    - Ele omite a maioria dos detalhes do design para fornecer uma visão geral da funcionalidade do sistema
    - Permite estudar implicações de diferentes alternativas de projeto
  - **Design** é tão **complexo** que é **preciso** um "**design**" abstrato simplificado para introduzir o design aos novos membros da equipe

Classe de Análise → Classe de Projeto

- Possíveis transformações
  - Uma classe de análise pode vir a ser uma única classe no modelo de design.
  - Uma classe de análise pode fazer parte de uma classe no modelo de design.
  - Uma classe de análise pode vir a ser uma classe agregada no modelo de design (ou seja, as partes dessa classe agregada podem não estar definidas explicitamente no modelo de análise).
  - Uma classe de análise pode vir a ser um grupo de classes que herda da mesma classe no modelo de design.
  - Uma classe de análise pode vir a ser um grupo de classes relacionadas funcionalmente no modelo de design.

http://www.wthreex.com/rup/portugues/index.htm

Classe de Análise → Classe de Projeto

- Possíveis transformações
  - Uma classe de análise pode vir a ser um pacote no modelo de design (ou seja, pode tornar-se um componente).
  - Uma classe de análise pode vir a ser um relacionamento no modelo de design.
  - Um relacionamento entre as classes de análise pode vir a ser uma classe no modelo de design.
  - As classes de análise lidam principalmente com requisitos funcionais e objetos de modelo provenientes do domínio "problema". As classes de design lidam com requisitos não-funcionais e objetos de modelo provenientes do domínio "solução".
  - As classes de análise podem ser usadas para representar "os objetos aos quais desejamos que o sistema dê suporte", sem que você precise definir quantas suportarão o hardware e quantas suportarão o software. Assim, parte de uma classe de análise pode ser usada pelo hardware e não estar definida no modelo de design.

http://www.wthreex.com/rup/portugues/index.htm

Referências

- Craig Larman. Utilizando UML e Padrões. Bookman. (2d edição mais enxuta, 3ed mais completa)
  - Capítulos 9-11 da edição 2
  - Capítulos 9 da edição 3

  - Descreve passo a passo um Processo de Análise e Projeto Orientados a Objetos utilizando a notação UML. Aborda também o uso de padrões de projeto.
  - As duas primeiras edições são mais objetivas e sucintas, a terceira é mais focada em desenvolvimento iterativo e ágil.

Perguntas?

- Este material tem contribuições de
  - Ingrid Nunes
  - Karin Becker
  - Lucinéia Thom
  - Marcelo Pimenta

Prosoft

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf
INSTITUTO DE INFORMÁTICA
UFRGS