Modelagem

- Entendimento do ambiente
- Permite lidar com complexidade
  $\rightarrow$ complexidade progressiva

Modelos são úteis para:

- Entendimento de problemas
- Comunicação entre stakeholders
- Compreensão dos requisitos
- Difundir conhecimento entre stakeholders
- Avaliar diferentes soluções

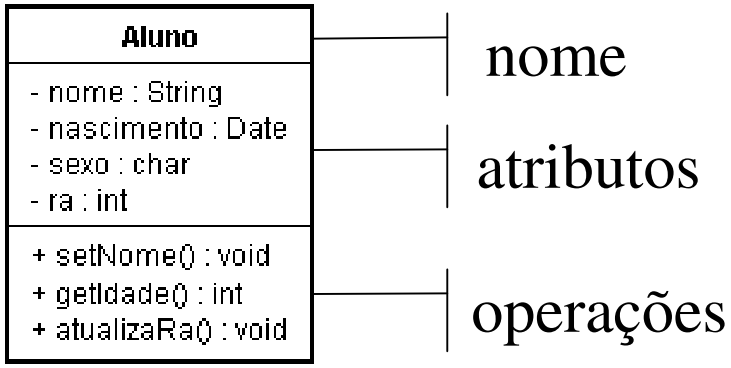
Objetivo de hoje:

- Entender conceitos por trás dos elementos e como representá-los nos diagramas de classes

- 3 elementos principais:
  - Classes
  - Atributos
  - Operações

Classe

- Classe é uma descrição de um conjunto de objetos que compartilham os mesmos atributos, operações, relacionamentos e semântica.



Atributos

- Ao definirmos atributos podemos determinar além de seu nome e tipo de dados, seu valor inicial, visibilidade e outras características.
- A única informação obrigatória é o próprio nome do atributo, mas conforme a modelagem é refinada, outras características do atributo tornam-se importantes para o projeto.
- Sintaxe :

  Visibilidade nome:tipo = valor inicial
- **Exemplos:**
  - nome:String = ‘ ‘
  - idade: integer = 0
  - peso: Double = 0.0
  - sexo: TipoSexo = ‘M’
  - estaLigado:Boolean = true

Operações

- A modelagem de operações também não se limita a seu nome e parâmetros.
- A única informação obrigatória é o nome da operação, mas conforme a modelagem é refinada, outras características da operação tornam-se importantes para o projeto.
- Sintaxe:

  Visibilidade nome (lista de parâmetros) tipo-de-retorno
- **Exemplos**:

  +exibirProduto()

  +verPromoção(Codigo:int):boolean

  +obterTotalVendas(vendedor):real
- **Assinatura da operação**
  - É como a operação é solicitada, é um conceito importante que é composto do nome da operação, seus parâmetros e o tipo de retorno.

Visibilidade

- Identifica como o elemento poderá ser visto externamente:

+ (público) - A propriedade será vista e usada dentro da classe na qual foi declarada, em qualquer elemento externo(incluindo objetos instanciados a partir desta classe) e nas classes descendentes.

 # (protegido) - A propriedade será vista e usada apenas dentro da classe na qual foi declarada e pelas classes descendentes.

- (privado) - A propriedade será vista e usada apenas dentro da classe na qual foi declarada.

~ (pacote) - A propriedade poderá ser vista e usada por elementos que estejam declarados dentro do mesmo pacote no qual está inserida a classe que a declarou. È como a visibilidade pública, só que não generalizada a qualquer elemento externo, mas apenas aos elementos localizados no mesmo pacote.

Visibilidade

- Para garantir o encapsulamento devemos definir os atributos como privados e criar métodos públicos que manipulem esses atributos.

Convenções

- Nome da classe inicia com letra maiúscula;
- Nome da classe no singular;
- Cada classe deve ter um nome (normalmente substantivo) que a diferencie de outras classes;
- Podemos ter classes sem atributos ou métodos;
- Nome do atributo inicia com letra minúscula, se for composto de dois termos o segundo deve iniciar com letra maiúscula. Ex: dataNascimento;

Exercícios

- Defina uma classe representando uma pessoa com os seguintes atributos: nome, ano de nascimento e altura (em metros). Defina operações para a iniciação desses atributos. Adicione uma operação que retorna a idade aproximada de uma pessoa de acordo com um determinado ano de entrada. Adicione outra operação que retorna sua altura em centímetros.

Exercícios

- Defina uma classe representando um ItemDeEstoque com os seguintes atributos: nível de estoque (inteiro) e preço(float). Defina operações para consultar os valores desses atributos e também para iniciá-los usando parâmetros. Adicione mais dois métodos para permitir a atualização do nível de estoque de forma apropriada (isto é, para baixa e para reposição do estoque)

Exemplo adicional

Suponha a existência de um banco hipotético onde as contas correntes se comportam do seguinte modo:

(1) As contas são sempre individuais e a única informação necessária a respeito do cliente é o seu nome.

(2) As contas são identificadas através de números.

(3) As contas devem receber um depósito inicial no momento de sua abertura.

(4) Uma vez criada uma conta, ela pode receber lançamentos de crédito e débito.

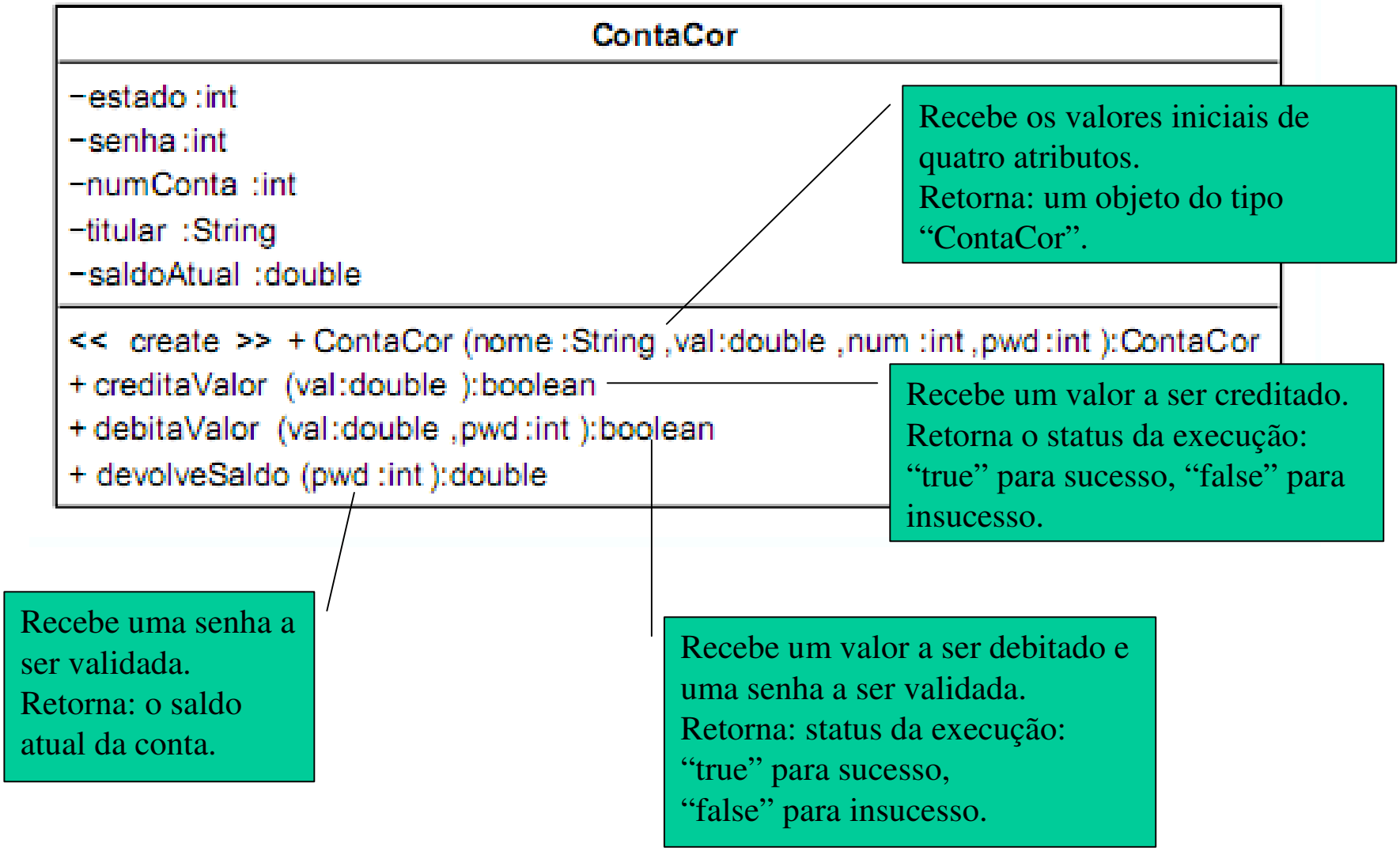
(5) O saldo de uma conta nunca pode ficar negativo.Qualquer pedido de saque superior ao saldo deve ser rejeitado.

(6) Se, durante a movimentação da conta, o saldo se igualar a zero, a conta torna-se inativa; não podendo mais ser reaberta.

(7) Deve ser prevista uma operação de consulta de saldo.

(8) As operações de débito e consultas só podem ser efetuadas se fornecida uma senha numérica validada pelo cliente. A senha da conta é determinada no momento de sua criação.

Solução


ContaCor
-estado :int
-senha :int
-numConta :int
-titular :String
-saldoAtual :double
<< create >> + ContaCor (nome :String ,val :double ,num :int ,pwd :int):ContaCor
+ creditaValor (val :double):boolean
+ debitaValor (val :double ,pwd :int):boolean
+ devolveSaldo (pwd :int):double
Recebe os valores iniciais de quatro atributos.
Retorna: um objeto do tipo "ContaCor".
Recebe um valor a ser creditado.
Retorna o status da execução:
"true" para sucesso, "false" para insucesso.
Recebe uma senha a ser validada.
Retorna: o saldo atual da conta.
Recebe um valor a ser debitado e uma senha a ser validada.
Retorna: status da execução:
"true" para sucesso,
"false" para insucesso.

Implementação da Classe

```
// arquivo ContaCor.java
class ContaCor {
private int estado; // 1 = ativo ; 2 = inativo
private int senha;
private int numConta;
private String titular;
private double saldoAtual;
public ContaCor(String nome, double val,
int num ,int pwd) { ...} }
public double devolveSaldo(int pwd) { ...}
public boolean creditaValor(double val) { ...}
public boolean debitaValor(double val, int pwd) { ...}
} // fim da classe ContaCor
```

Implementação do Comportamento

```
public ContaCor(String nome, double val,
int num, int pwd) {
titular = nome;
numConta = num;
senha = pwd;
saldoAtual = val;
estado=1; //conta é ativada quando criada
} // fim do construtor
```

Implementação do Comportamento

```
public double devolveSaldo (int pwd) {
if (estado!=1) return(-1); // conta deve estar ativa
if (pwd!=senha) return(-1); //senha deve ser valida
return(saldoAtual);
} // fim de devolveSaldo( )
```

Implementação do Comportamento

```
public boolean creditaValor (double val) {
if (estado!=1)
return(false); //conta deve estar ativa
if (val<=0)
return (false); // val>0;
saldoAtual+= val; // credita valor;
return(true); // operação terminada com
   sucesso
} // fim de creditaValor( )
```

Implementação do Comportamento

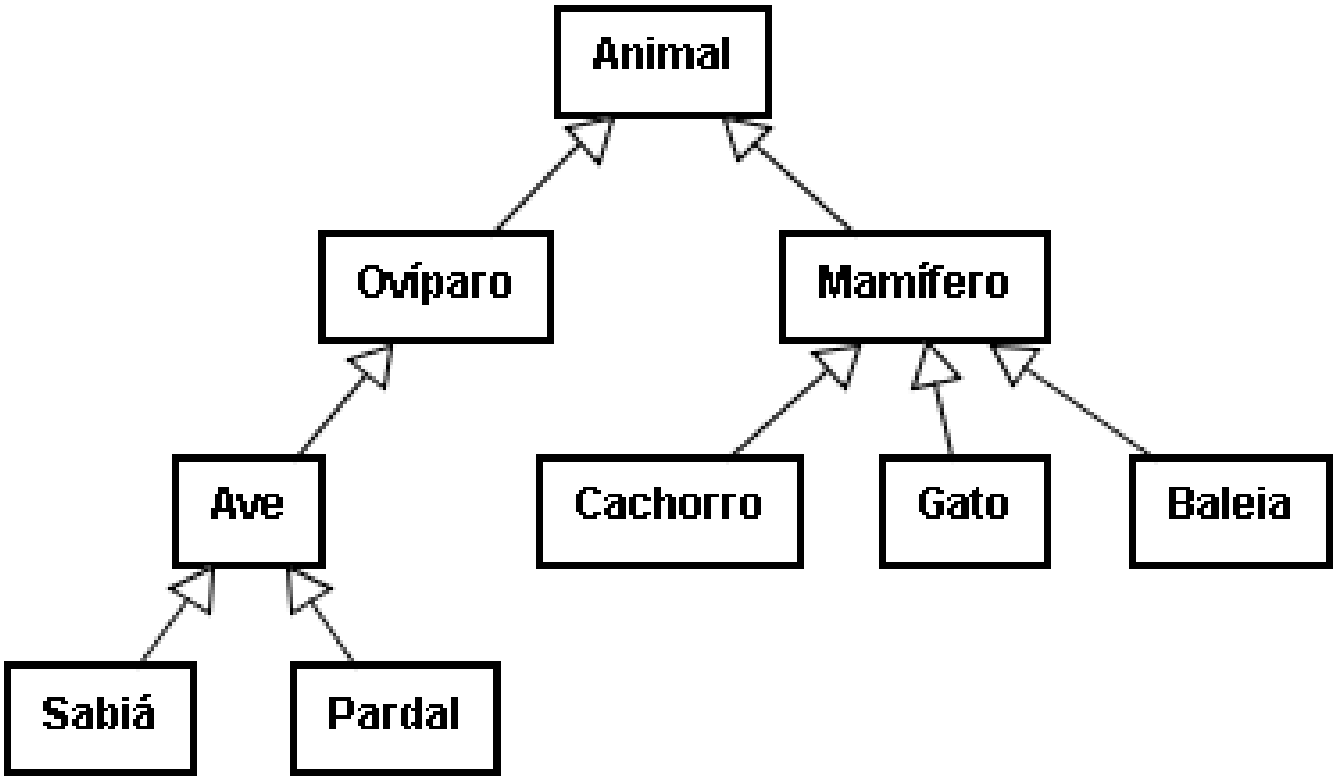
```
public boolean debitaValor (double val, int pwd) {
if (estado!=1)
    return(false); // conta deve estar ativa
if (val<=0)
    return (false); // val>0;
if (pwd!=senha)
    return (false);//senha deve ser valida
if (val>saldoAtual)
    return (false);// val<= saldoAtual
saldoAtual -= val; //debita valor
if(saldoAtual = =0)
estado=2;// se saldo=0, torna conta inativa
return(true);
} // fim de debitaValor( )
```

Classe

- Classe Concreta:
  - pode ser instanciada
- Classe Abstrata:
  - não pode ser instanciada
  - seu nome deve ser escrito em itálico

Exercício

- Dada a hierarquia de animais apresentada a seguir, quais as classes que poderiam ser classificadas como abstratas? Justifique sua resposta.



Relacionamento

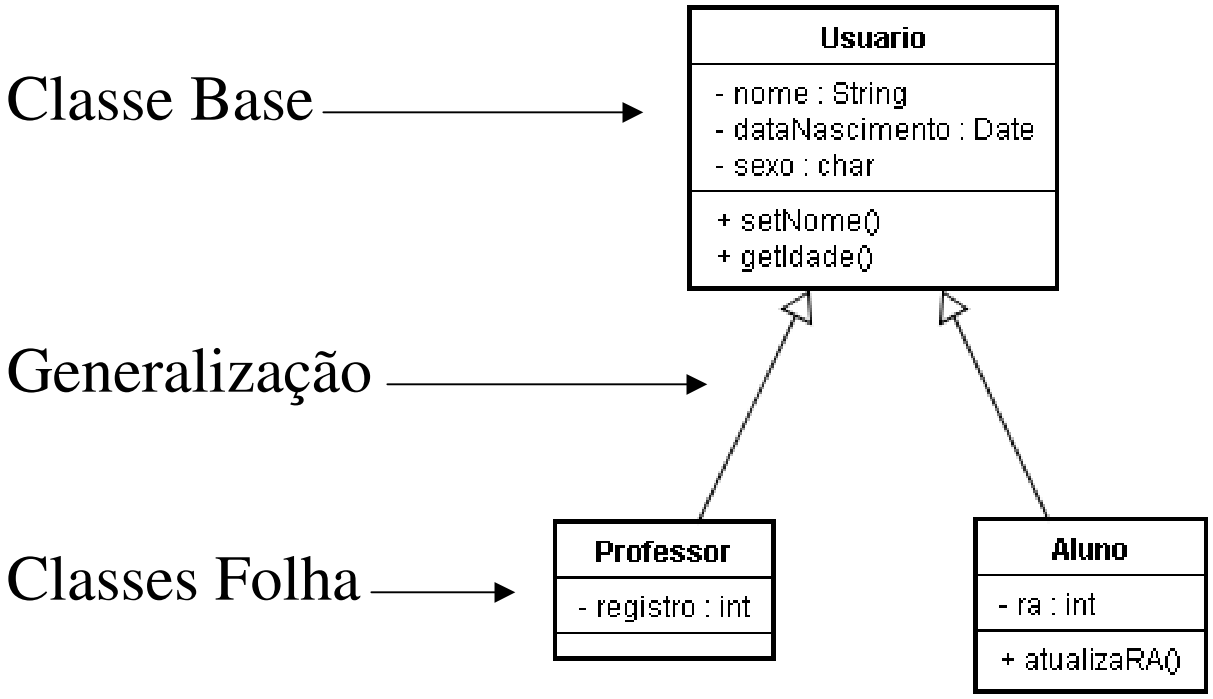
- Especifica como as classes podem se comunicar, como pode haver troca de mensagem;
- Tipos de relacionamento:
  - Dependência: usa
  - Generalização: tipo
  - Associação
    - Associação
    - Agregação: faz parte
    - Composição: o todo

Relacionamento – Generalização

- Generalização ou herança define um relacionamento onde uma classe (filha – subclasse - folha) herda todas as características (atributos, operações e relacionamentos) de outra classe (pai – superclasse - base).
- As subclasses podem ter características adicionais que só se aplicam ao seu nível de hierarquia.
- Não são nomeados os relacionamentos de herança e não se aplica multiplicidade

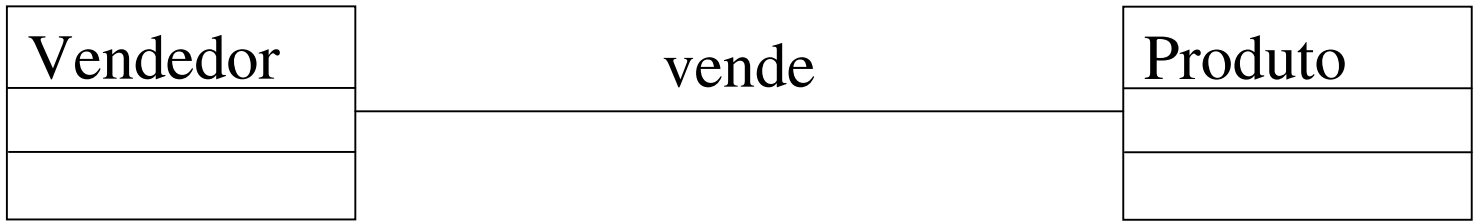
Generalização - Herança

- "É um tipo de"



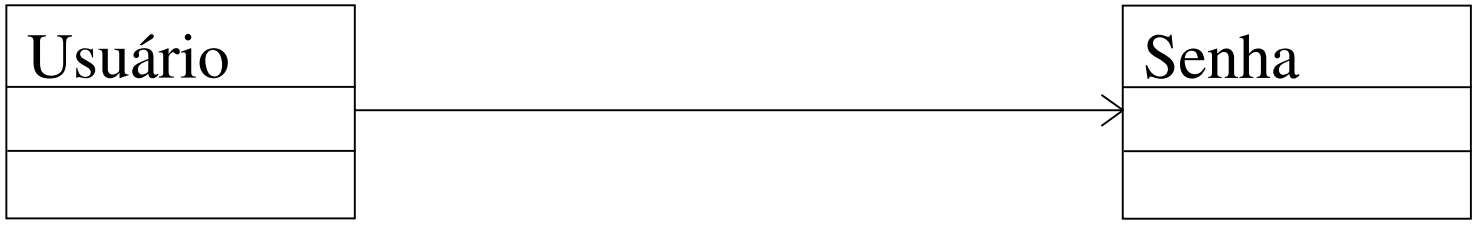
Associação

- Conexão bidirecional entre classes
- Indica a existência de uma ligação entre objetos das classes associadas



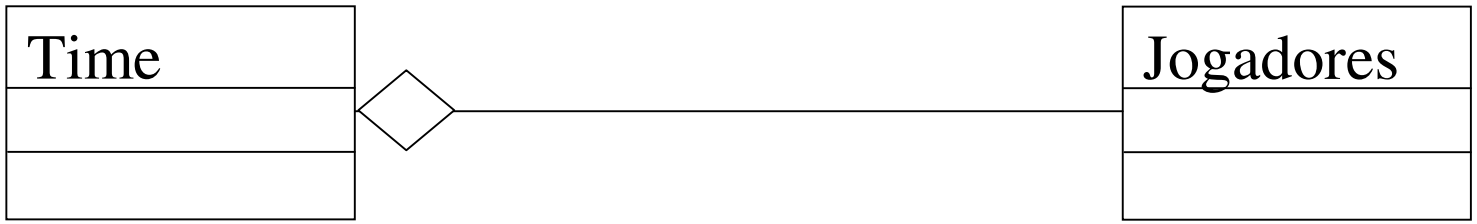
Associação

- Navegação
  - Se não for especificada, é bidirecional, ou seja, de um objeto de qualquer uma das classes é possível navegar até um objeto da outra classe.
  - Para indicar a direção da navegação, incluímos setas indicando a direção da navegação
  - Um Usuário sabe quais são as senhas, mas senha não sabe a qual usuário ela está associada.



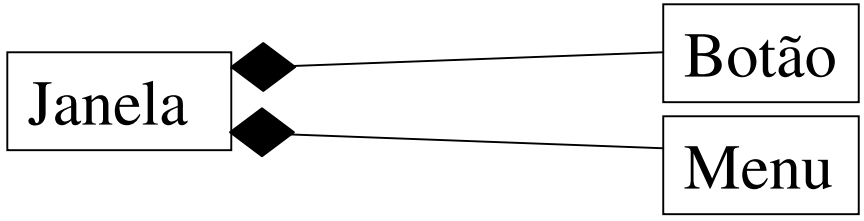
Agregação

- É uma forma especial de associação onde uma classe faz parte da outra.
- Para identificar se é agregação ou não:
  - A frase “parte de” é usada para descrever o relacionamento?
  - Existem algumas operações no todo, automaticamente aplicada às suas partes?
- Notação



Composição

- É um relacionamento forte de agregação onde se o objeto todo for destruído suas partes também serão.
- Exemplo:
  - Se eu acabar com o Time eu posso ter ainda os jogadores jogando para outros times, então a relação é agregação simples.
  - Imagine uma Janela de uma aplicação, que tenha Botão, Menu. Se você destruir a janela todas as partes serão destruídas, então a relação é uma composição.



Exercícios

- Considere como domínio do problema um zoológico onde existem diversos tipos de animais com uma estrutura administrativa, como por exemplo, tratadores, veterinários, etc...
- A administração do zoo envolve o preparo de cardápios específicos para cada tipo de bicho, assim, o sistema deve possibilitar o cadastro dos animais existentes no zoo, classificados de acordo com a respectiva classe: mamífero, réptil ou ave.

Exercício

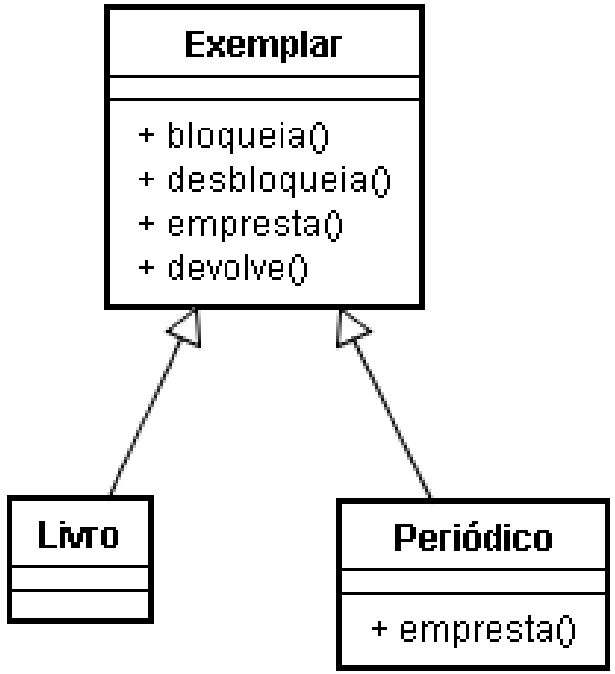
- Um editor de desenhos geométricos possui uma tela para desenhar. Essa tela pode ser constituída de muitas figuras. Figuras podem ser círculos, linhas, polígonos e grupos. Um grupo consiste de muitas figuras e um polígono é composto por um conjunto de linhas. Quando um cliente pede para que uma tela se desenhe, a tela, por sua vez, pede para cada uma das figuras associadas que se desenhe. Da mesma forma, um grupo pede que todos os seus componentes se desenhem. Crie uma hierarquia de generalização/especialização que classifique essas diferentes figuras geométricas e identifique o comportamento de cada classe que você criar, bem como os seus respectivos atributos.

Exercícios

- Considere como domínio do problema uma universidade como a UNICAMP, onde existem diversas unidades, que podem ser institutos, faculdades e núcleos. Unidades são subdivididas em departamentos. Uma universidade tem um quadro de pessoal permanente composto por funcionários, que podem ser professores, secretários, analistas, copeiras, etc...
- Nos cursos oferecidos pelas faculdades ingressam alunos de graduação, pós-graduação e de extensão universitária. Cursos são compostos por disciplinas de acordo com o catálogo da universidade. Alunos podem se matricular em turmas de uma disciplina específica. A pós-graduação pode ter programas de mestrado acadêmico, mestrado profissional e de doutorado.

Exercício

- Considere a hierarquia de classes concretas, usada por um sistema de controle de uma biblioteca. O sistema tem que ser evoluído para acomodar novas funcionalidades. Considere que a biblioteca agora tenha que manter fitas de vídeo no seu acervo, para as quais estão previstos os requisitos:
  - Cada fita tem associada a ela um tipo de privilégio, a saber: acesso permitido apenas a professores, acesso permitido a alunos e professores e acesso para qualquer usuário.
  - O tipo de privilégio pode ser alterado a qualquer instante
  - O prazo de emprestimo é sempre 2 dias
  - As fitas não podem ser bloqueadas/desbloqueadas.
- Altere a hierarquia de classes original, utilizando o conceito de classes abstratas, de tal forma que os requisitos acima descritos sejam incorporados no diagrama de classes.

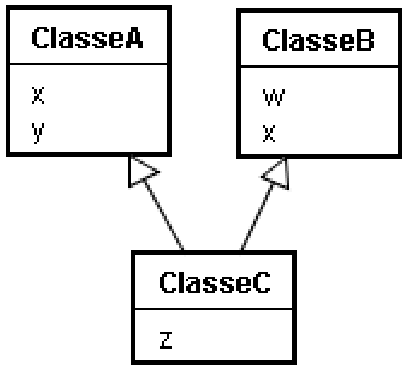


Exercício

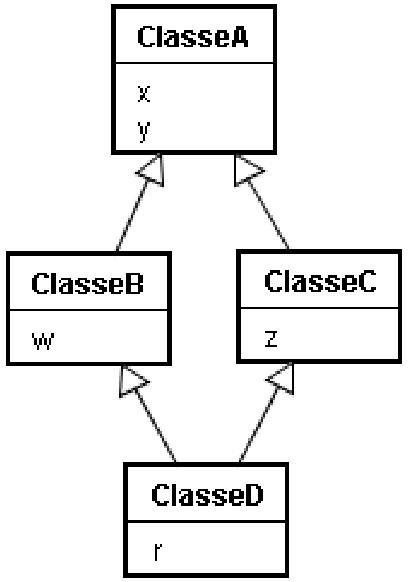
- Suponha que o sistema sofra uma nova mudança, passando a controlar também o empréstimo de salas de estudo e CDs com cópias de software livre. O empréstimo das salas é efetuada por um período de no máximo 2 horas. As salas podem ser emprestadas apenas para professores ou alunos e não podem ser bloqueadas/desbloqueadas. As regras para o empréstimo dos CDs são as mesmas das fitas de vídeo. Incorpore essas mudanças também.

Herança

- Herança Simples: quando uma subclasse herda características de uma única superclasse
- Herança Múltipla: quando uma classe é definida a partir de duas ou mais superclasses;
- Problemas:

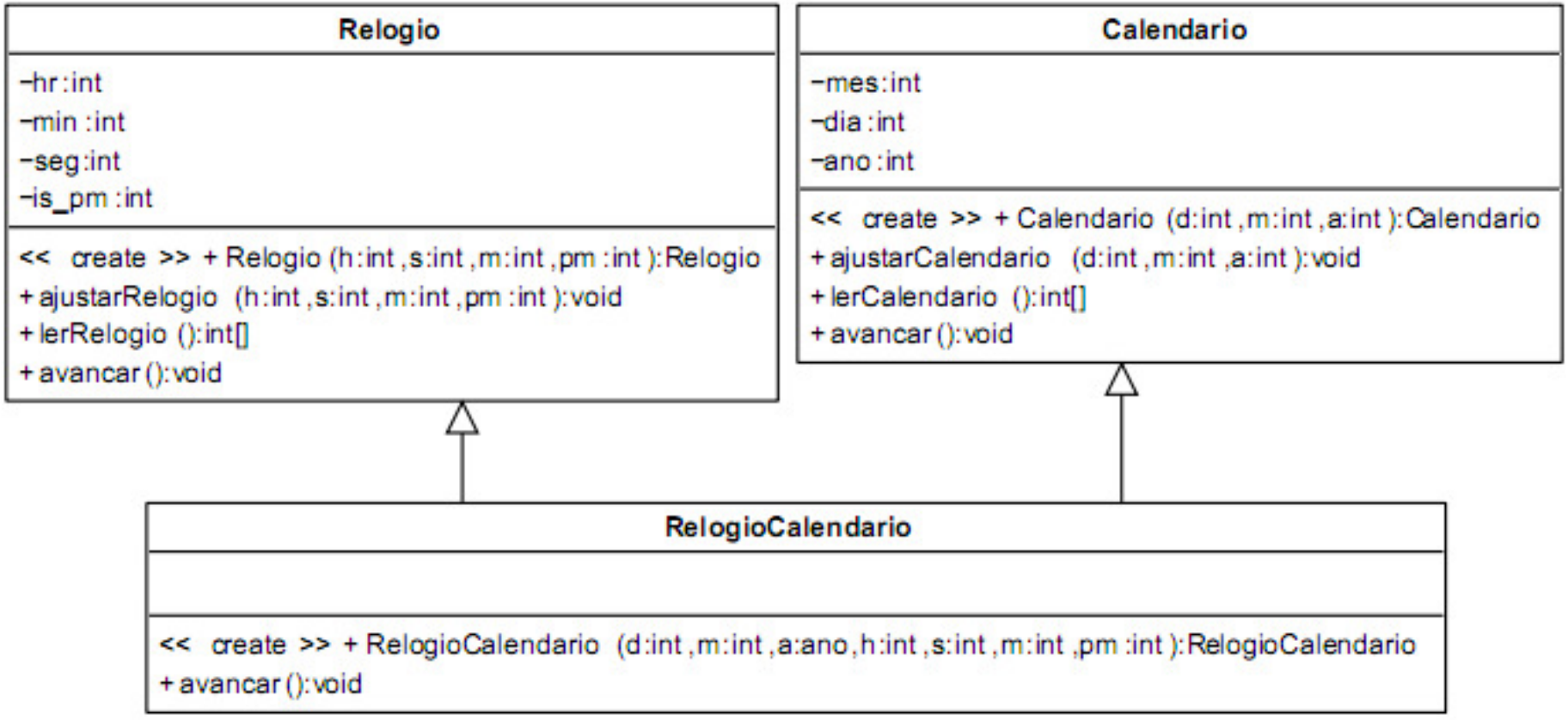


C herda x de A e de B
teremos colisão de nomes

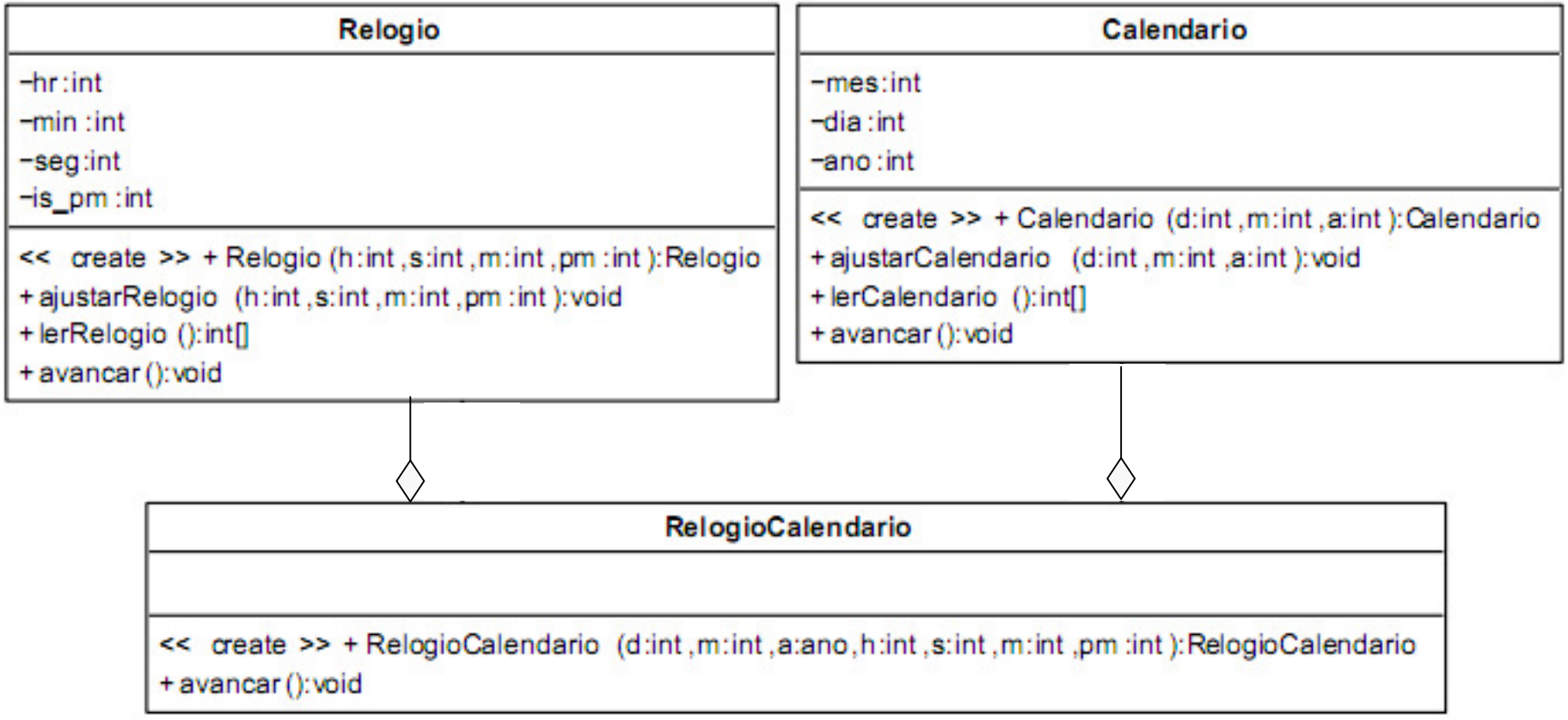


B e C herdam x e y e D herda dos dois?
teremos herança repetida

Exemplo Herança Múltipla

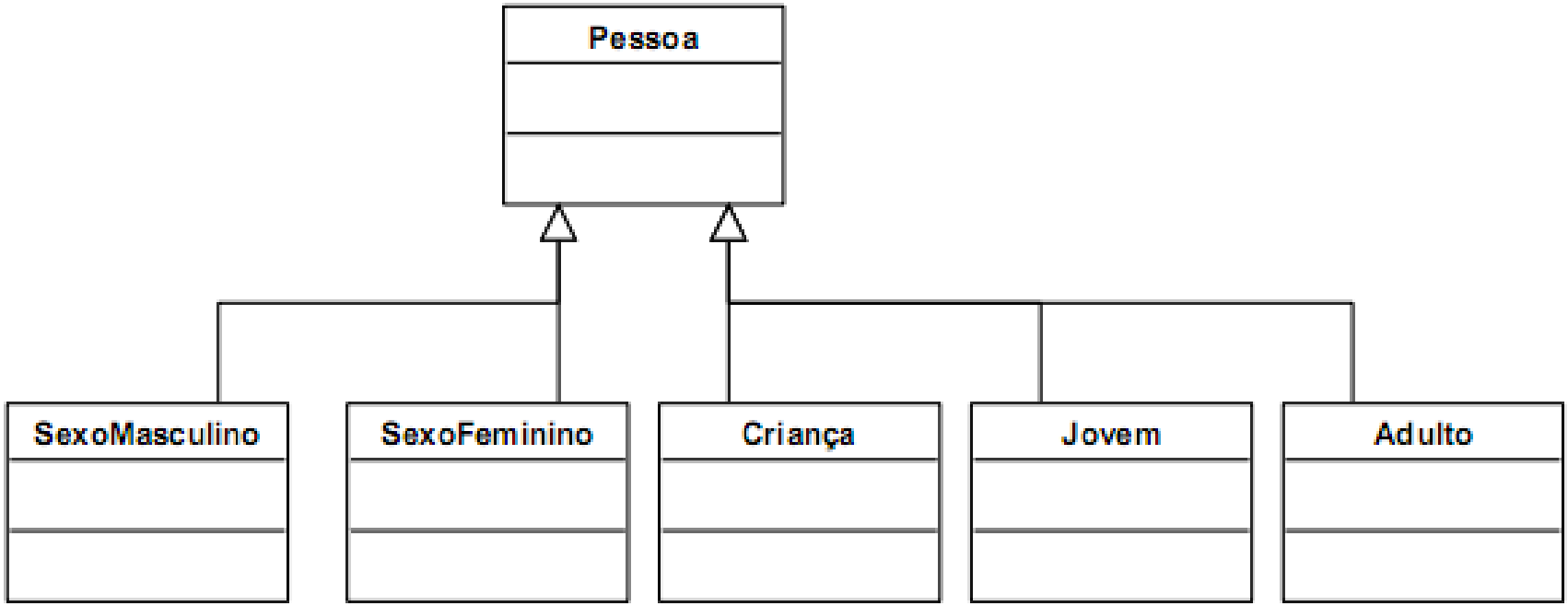


Solução Herança Múltipla



Exercício

- Uma pessoa pode ser classificada de acordo com o seu sexo como masculino ou feminino e pode também ser classificada de acordo com a sua idade como criança, jovem e adulto.



Exercício

- Apresente uma solução de modelagem que combine essas duas perspectivas usando herança múltipla. Note que um objeto pode ser instanciado a partir de apenas uma única classe (obs. um objeto não pode ser instância de 2 classes).
- Por exemplo, o objeto José é do sexo masculino e adulto, mas ele não pode ser instanciado a partir de ambas as classes SexoMasculino e Adulto ao mesmo tempo.

Exercício

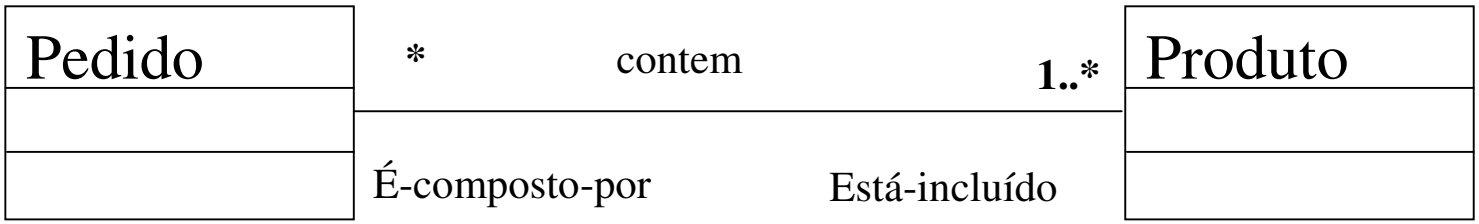
Proponha uma solução alternativa mais flexível que supere as desvantagens identificadas no item anterior. (Dica: pense na hierarquia de agregação/decomposição.)

Indicador de Multiplicidade

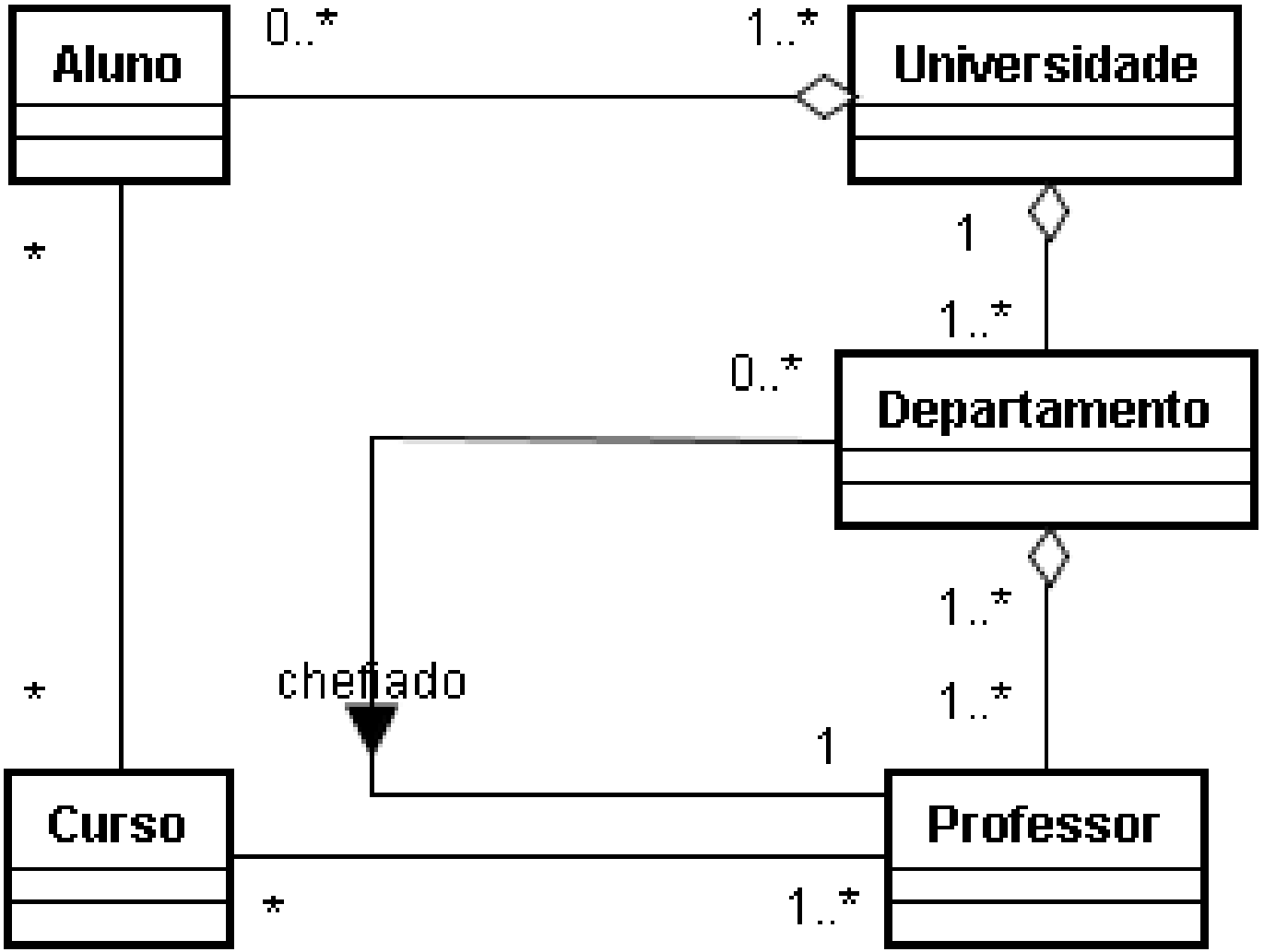
- Embora a multiplicidade seja especificada para classes, ela define a quantidade de objetos que participam de um relacionamento.
  - Exatamente um **1**
  - Muitos (0 ou mais) *****
  - Obrigatório (1 ou mais) **1..***
  - Opcional (0 ou 1) **0..1**
  - Especificado numericamente **1-3,5**
  - Papel ordenado **(ordered)***

Indicador de Multiplicidade

- Pedido está associado a no mínimo 1 e no máximo muitos produtos
- Produto está associado a 0 ou muitos Pedidos

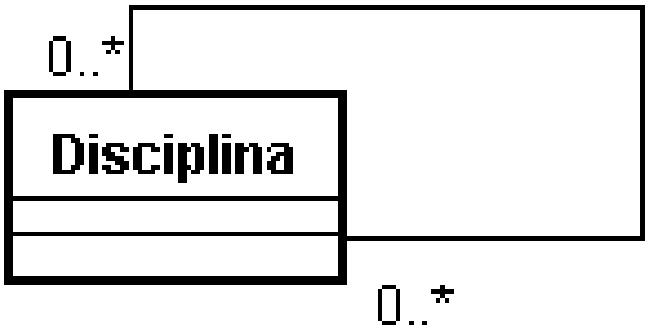


Exemplo


Aluno
0..*
1..*
Universidade
*
1
1..*
0..*
Departamento
1..*
*
chefiado
1..*
1
Curso
Professor
*
1..*

Relacionamento Reflexivo

- Objetos pertencentes à mesma classe podem precisar comunicar-se uns com os outros.
- Exemplo: Temos uma grade de disciplinas de um curso e algumas delas precisam de pré-requisitos para serem cursadas.



Polimorfismo

- Redefinição de Operações - É um caso especial de polimorfismo de inclusão, quando uma classe define uma forma de implementação especializada para um método herdado da superclasse.
- Redefinição :
  - a nova operação deve ter a mesma assinatura da operação herdada, isto é, elas devem ser idênticas quanto ao nome da operação e a lista de parâmetros (mesmo número de parâmetros, com os mesmos tipos e declarados na mesma ordem)
- Sobrecarga:
  - ocorre quando existe apenas coincidências nos nomes das operações, isto é, as listas de parâmetros não são idênticas, não tem mesma assinatura.

Redefinição x Sobrecarga

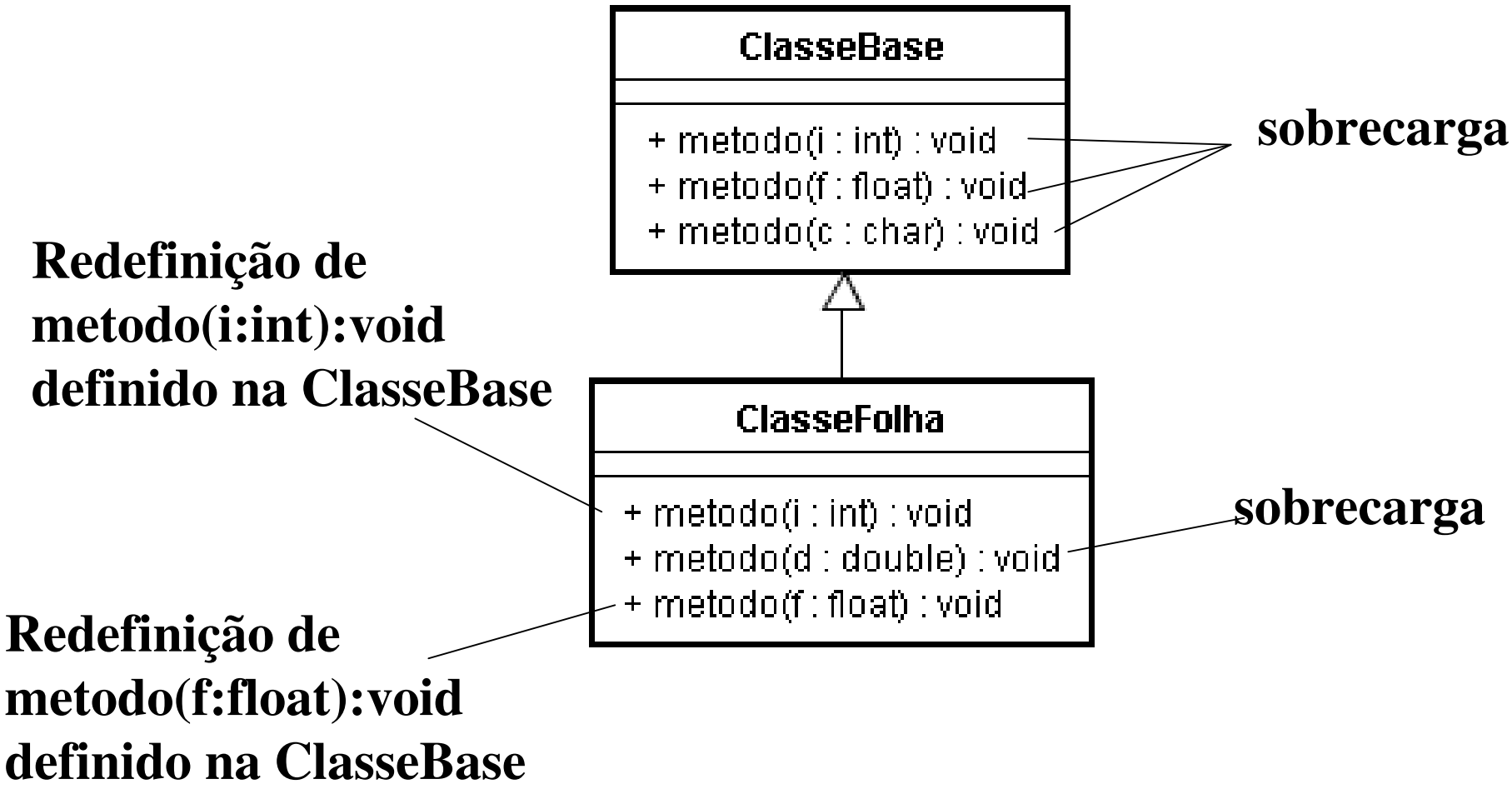
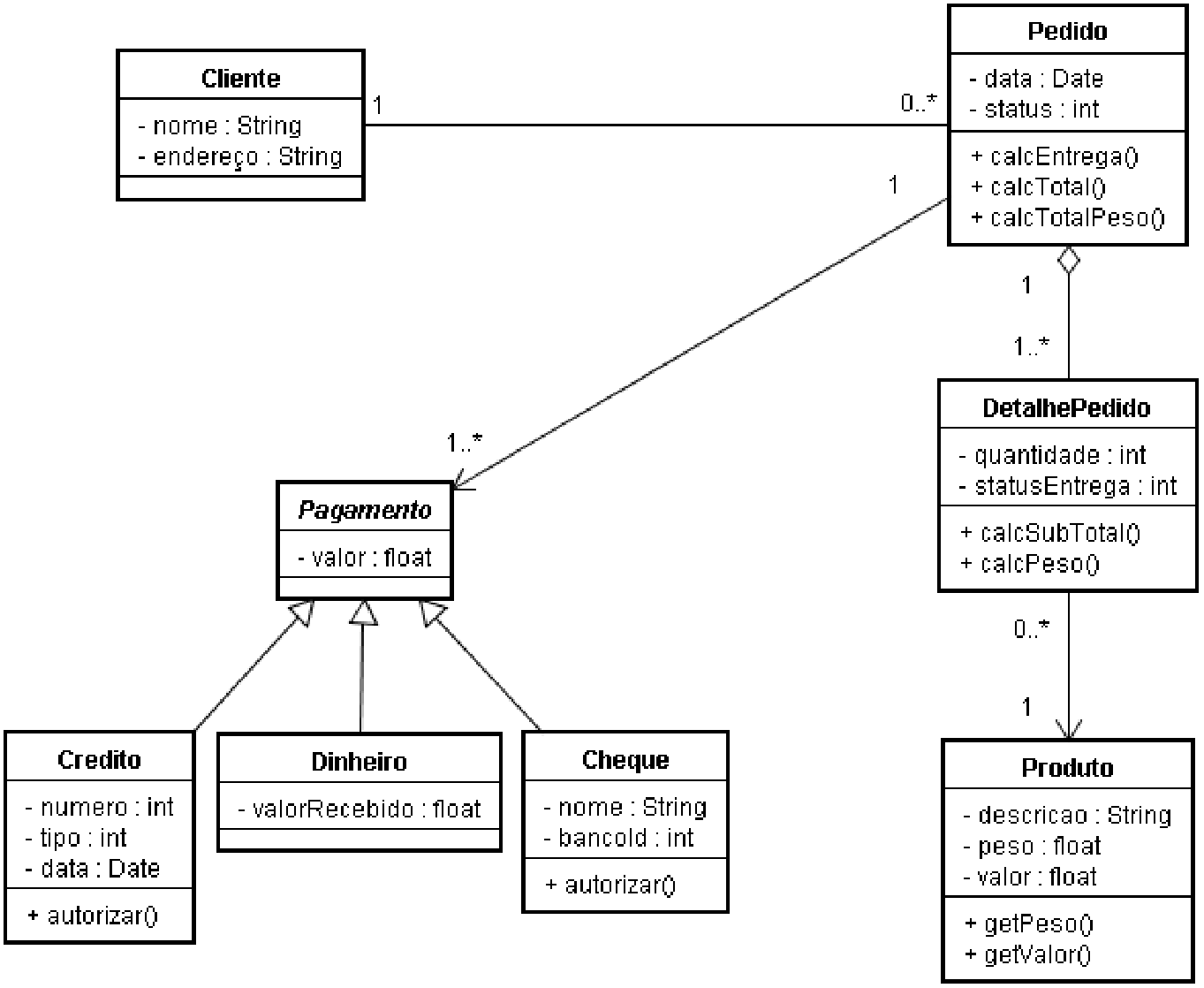


Diagrama de Classe - Objetivos

- Modelar a visão estática do projeto de um sistema
  - Identificar conceitos presentes no sistema - classes
- Modelar o relacionamento entre objetos que farão parte de um sistema
  - Associação, agregação, composição, herança
- Capturar o vocabulário do Problema;
- Ancorar os comportamentos em suas classes
  - Que classe é responsável por qual operação
- Gerar as declarações das estruturas das classes

Exemplo


Cliente
- nome : String
- endereço : String
1
0..*
Pedido
- data : Date
- status : int
+ calcEntrega()
+ calcTotal()
+ calcTotalPeso()
1
1..*
1
1..*
DetalhePedido
- quantidade : int
- statusEntrega : int
+ calcSubTotal()
+ calcPeso()
Pagamento
- valor : float
0..*
1
Credito
- numero : int
- tipo : int
- data : Date
+ autorizar()
Dinheiro
- valorRecebido : float
Cheque
- nome : String
- bancoId : int
+ autorizar()
Produto
- descricao : String
- peso : float
- valor : float
+ getPeso()
+ getValor()

Seqüência de atividades sugeridas

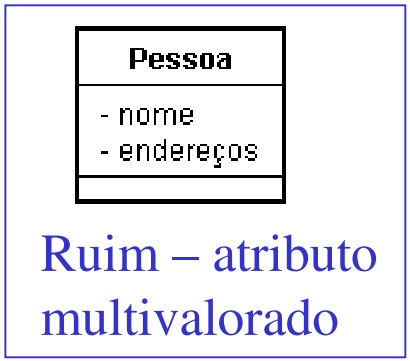
- Identifique um conjunto de classes inicial;
- Adicione atributos e comportamentos;
- Encontre generalizações;
- Adicione associações e multiplicidade;
- Reveja o modelo construído adicionando/removendo classes, atributos, comportamentos, generalizações ou associações.

Identificando Classe e seus Atributos

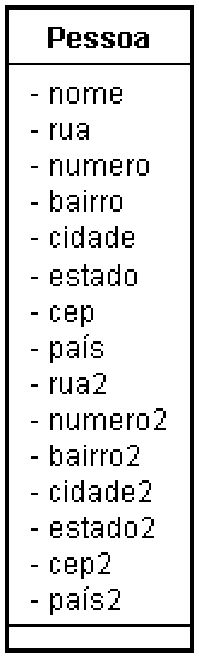
- Ler a especificação de requisitos
- Extrair nomes (substantivos): eles serão classes ou atributos
- Eliminar os que são:
  - Redundantes;
  - Representem instâncias;
  - São vagos ou genéricos demais;
  - Desnecessários para a solução do problema.
- Estar atento para nomes que representam atores que interagem com o sistema. Eles poderão ser classes ou não dependendo da necessidade de cadastra-los no sistema.

Identificando Classe e seus Atributos

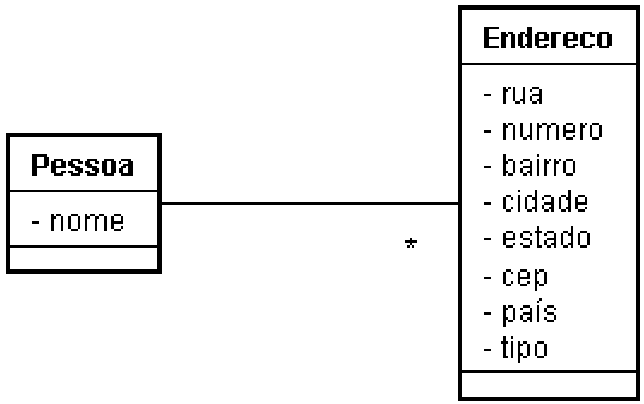
- Se um subconjunto de atributos de uma classe formam um grupo coerente, pode ser interessante criar uma outra classe para esses atributos. É útil também quando esse grupo é comuns a várias classes



Ruim – atributo multivalorado



Ruim – muitos atributos e só podem ter dois endereços



Boa solução: o atributo tipo indica se é endereço residencial, comercial...

Identificando Comportamento

- Ler a especificação de requisitos;
- Extrair verbos ou algo que signifique alguma ação no contexto do sistema;
- Eliminar os que são:
  - Redundantes;
  - São vagos ou genéricos demais;
  - Desnecessários para a solução do problema.
- Colocar os comportamentos extraídos nas classes especificadas.

Identificando Generalizações

- Duas formas de identificação:
  - Bottom-up: agrupar classes semelhantes criando uma superclasse
  - Top-down: procurar por classes genéricas, para então, se necessário, especializá-las

Identificando Associações

- Iniciar com as classes consideradas mais importantes;
- Decidir quais os relacionamentos destas com as outras.
  - Uma associação existe se uma classe:
    - Possui
    - Controla
    - Está conectada
    - Está relacionada
    - É parte de
    - Tem como parte
    - É membro de
    - Tem como membro

  Alguma outra classe no modelo.

Identificando Associações

- Evite adicionar muitas associações em uma classe pois o acoplamento fica muito alto.
- Defina a multiplicidade.

Estudo de Caso -Locadora de Veículos

- Uma locadora de veículos deseja um sistema para facilitar o atendimento a seus clientes. O processo de aluguel de carros atual é confuso e está gerando insatisfação entre os clientes.
- A locadora é composta basicamente pelos seus funcionários e carros para aluguel. Os funcionários são identificados por cpf, nome, endereço, telefone.
- Já os carros estão divididos em diversos tipos: popular, luxo, utilitário, etc. As informações importantes sobre os carros a serem armazenadas são: código (chapa do carro), tipo, modelo, ano, cor, chassis, km e valor do aluguel (diárias e semanais).
- Os funcionários serão responsáveis pelo cadastro dos clientes e dos carros adquiridos pela locadora, por efetuar o aluguel de um carro para o cliente e dar baixa no aluguel.
- Existem clientes especiais e clientes comuns. Os especiais possuem uma taxa de desconto e um valor de quilometragem extra para seus aluguéis. Qualquer cliente é identificado por rg, nome, cpf, telefone, endereço, cidade.
- Desta forma, o cliente poderá solicitar o aluguel de carros a um funcionário da locadora.

Estudo de Caso -Locadora de Veículos

- Os tópicos abaixo descrevem as funcionalidades do sistema.
  - Alugar Carro: cliente deve solicitar ao funcionário o aluguel do carro. O sistema verifica se o carro solicitado pelo cliente está disponível. Caso esteja, o processo de locação é concluído e o carro passa a estar indisponível. A data de aluguel deve ser guardada para calculo do valor do aluguel na devolução.
  - Dar Baixa: cliente faz devolução do carro para o funcionário e solicita nota fiscal (recibo) com a quilometragem percorrida e o valor do aluguel. O funcionário coloca o status do carro novamente como disponível, solicita ao sistema para calcular o valor a ser pago e emite o recibo para o cliente.
  - Cadastrar Cliente: cliente solicita ao funcionário que o cadastre na locadora. O funcionário recebe os dados e cadastra-o.
  - Cadastrar Carro: funcionário cadastra o carro adquirido.

Identificando Substantivos:

- Uma locadora de veículos deseja um sistema para facilitar o atendimento a seus clientes. O processo de aluguel de carros atual é confuso e está gerando insatisfação entre os clientes.
- A locadora é composta basicamente pelos seus funcionários e carros para aluguel. Os funcionários são identificados por cpf, nome, endereço, telefone.
- Já os carros estão divididos em diversos tipos: popular, luxo, utilitário, etc. As informações importantes sobre os carros a serem armazenadas são: código (chapa do carro), tipo, modelo, ano, cor, chassis, km e valor do aluguel (diárias e semanais).
- Os funcionários serão responsáveis pelo cadastro dos clientes e dos carros adquiridos pela locadora, por efetuar o aluguel de um carro para o cliente e dar baixa no aluguel.
- Existem clientes especiais e clientes comuns. Os especiais possuem uma taxa de desconto e um valor de quilometragem extra para seus aluguéis. Qualquer cliente é identificado por rg, nome, cpf, telefone, endereço, cidade.
- Desta forma, o cliente poderá solicitar o aluguel de carros a um funcionário da locadora.

Identificando Substantivos:

- Os tópicos abaixo descrevem as funcionalidades do sistema.
  - Alugar Carro: cliente deve solicitar ao funcionário o aluguel do carro. O sistema verifica se o carro solicitado pelo cliente está disponível. Caso esteja, o processo de locação é concluído e o carro passa a estar indisponível. A data de aluguel deve ser guardada para calculo do valor do aluguel na devolução.
  - Dar Baixa: cliente faz devolução do carro para o funcionário e solicita nota fiscal (recibo) com a quilometragem percorrida e o valor do aluguel. O funcionário coloca o status do carro novamente como disponível, solicita ao sistema para calcular o valor a ser pago e emite o recibo para o cliente.
  - Cadastrar Cliente: cliente solicita ao funcionário que o cadastre na locadora. O funcionário recebe os dados e cadastra-o.
  - Cadastrar Carro: funcionário cadastra o carro adquirido.

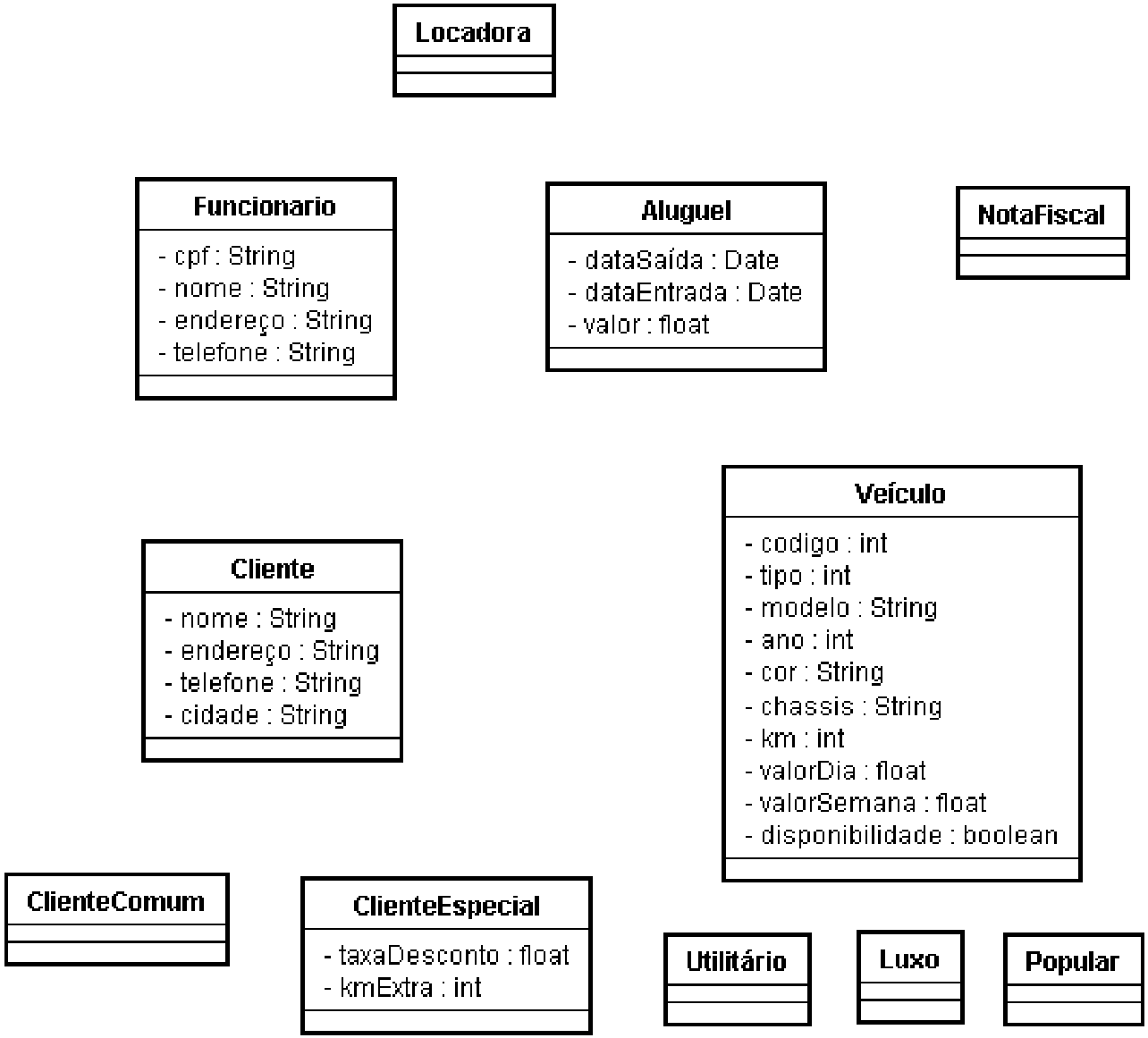
Lista de Substantivos

- locadora
- veículos
- clientes
- aluguel
- carros
- funcionários
- cpf
- nome
- endereço
- telefone.
- popular
- luxo
- utilitário
- código (chapa do carro)
- tipo
- modelo
- ano
- cor
- chassis
- km
- valor do aluguel diário
- valor do aluguel semanal
- clientes especiais
- clientes comuns
- taxa de desconto
- valor de quilometragem extra
- rg
- nome
- cpf
- telefone
- endereço
- cidade
- disponível
- indisponível
- data de aluguel
- valor do aluguel
- nota fiscal
- status

Classificando de Substantivos

- Classes
  - Locadora
  - Veículo
  - Cliente
  - Aluguel
  - Funcionário
  - Popular
  - Luxo
  - Utilitário
  - Cliente especial
  - Cliente Comum
  - Nota Fiscal
- Atributos
  - cpf (func.)
  - nome (func.)
  - endereço (func.)
  - telefone (func.)
  - código (veíc.)
  - tipo (veíc.)
  - modelo (veíc.)
  - ano (veíc.)
  - cor (veíc.)
  - chassis (veíc.)
  - km (veíc.)
  - valor aluguel diário (veíc.)
  - valor aluguel semanal (veíc.)
  - taxa desconto (Cliente esp)
  - valor km extra(Cliente esp)
  - nome (cliente)
  - endereço (cliente)
  - telefone (cliente)
  - cidade (cliente)
  - disponíbilidade (veíc.)
  - data aluguel(aluguel)
  - valor aluguel(aluguel)
- Descartados
  - Carro (sinônimo de Veículo)
  - status (= disponibilidade)

Modelando classes e atributos



Identificando Possíveis Operações:

- Uma locadora de veículos deseja um sistema para facilitar o atendimento a seus clientes. O processo de aluguel de carros atual é confuso e está gerando insatisfação entre os clientes.
- A locadora é composta basicamente pelos seus funcionários e carros para aluguel. Os funcionários são identificados por cpf, nome, endereço, telefone.
- Já os carros estão divididos em diversos tipos: popular, luxo, utilitário, etc. As informações importantes sobre os carros a serem armazenadas são: código (chapa do carro), tipo, modelo, ano, cor, chassis, km e valor do aluguel (diárias e semanais).
- Os funcionários serão responsáveis pelo cadastro dos clientes e dos carros adquiridos pela locadora, por efetuar o aluguel de um carro para o cliente e dar baixa no aluguel.
- Existem clientes especiais e clientes comuns. Os especiais possuem uma taxa de desconto e um valor de quilometragem extra para seus aluguéis. Qualquer cliente é identificado por rg, nome, cpf, telefone, endereço, cidade.
- Desta forma, o cliente poderá solicitar o aluguel de carros a um funcionário da locadora.

Identificando Possíveis Operações:

- Os tópicos abaixo descrevem as funcionalidades do sistema.
  - Alugar Carro: cliente deve solicitar ao funcionário o aluguel do carro. O sistema verifica se o carro solicitado pelo cliente está disponível. Caso esteja, o processo de locação é concluído e o carro passa a estar indisponível. A data de aluguel deve ser guardada para calculo do valor do aluguel na devolução.
  - Dar Baixa: cliente faz devolução do carro para o funcionário e solicita nota fiscal (recibo) com a quilometragem percorrida e o valor do aluguel. O funcionário coloca o status do carro novamente como disponível, solicita ao sistema para calcular o valor a ser pago e emite o recibo para o cliente.
  - Cadastrar Cliente: cliente solicita ao funcionário que o cadastre na locadora. O funcionário recebe os dados e cadastra-o.
  - Cadastrar Carro: funcionário cadastra o carro adquirido.

Possíveis operações identificadas

- aluguel de veículos
- cadastro dos cliente
- cadastro de veículo
- efetuar o aluguel de veículo
- dar baixa no aluguel
- solicitar o aluguel de carros a um funcionário da locadora
- alugar carro
- verifica se o carro solicitado pelo cliente está disponível
- carro passa a estar indisponível
- calculo do valor do aluguel
- dar baixa
- devolução do carro
- solicita nota fiscal
- coloca o status do carro novamente como disponível
- emite o recibo
- calcular o valor a ser pago
- cadastrar cliente
- cadastrar carro

Renomeando operações identificadas

- aluguel de veículos — alugar veículo
- cadastro dos cliente — cadastrar cliente
- cadastro de veículo — cadastrar veículo
- efetuar o aluguel de veículo — alugar veículo
- dar baixa no aluguel — devolver veículo
- solicitar o aluguel de carros a um funcionário da locadora
- alugar carro
- verifica se o carro solicitado pelo cliente está disponível — verificar disponibilidade
- carro passa a estar indisponível — alterar disponibilidade
- calculo do valor do aluguel — calcular valor aluguel
- dar baixa — devolver veículo
- devolução do carro — devolver veículo
- solicita nota fiscal — emitir nota fiscal
- coloca o status do carro novamente como disponível — alterar disponibilidade
- emite o recibo — emitir nota fiscal
- calcular o valor a ser pago — calcular valor aluguel
- cadastrar cliente
- cadastrar carro

Analisando operações identificadas

- Operações Identificadas
- alugar veículo
- cadastrar cliente
- cadastrar veículo
- dar baixa
- verificar disponibilidade
- alterar disponibilidade
- calcular valor aluguel
- emitir nota fiscal

- Operações Descartadas
- efetuar aluguel do veículo (repetição)
- solicitar o aluguel de carros a um funcionário da locadora (cliente conversa com funcionário - fora do contexto do sistema)
- alugar carro (sinônimo)
- devolver veículo(sinônimo)
- cadastrar carro (sinônimo)
- Colocar status do carro novamente como disponível (repetição)
- emitir recibo (sinônimo)
- calcular valor a ser pago (sinônimo)
- cadastrar cliente (repetição)
- cadastrar carro(repetição)

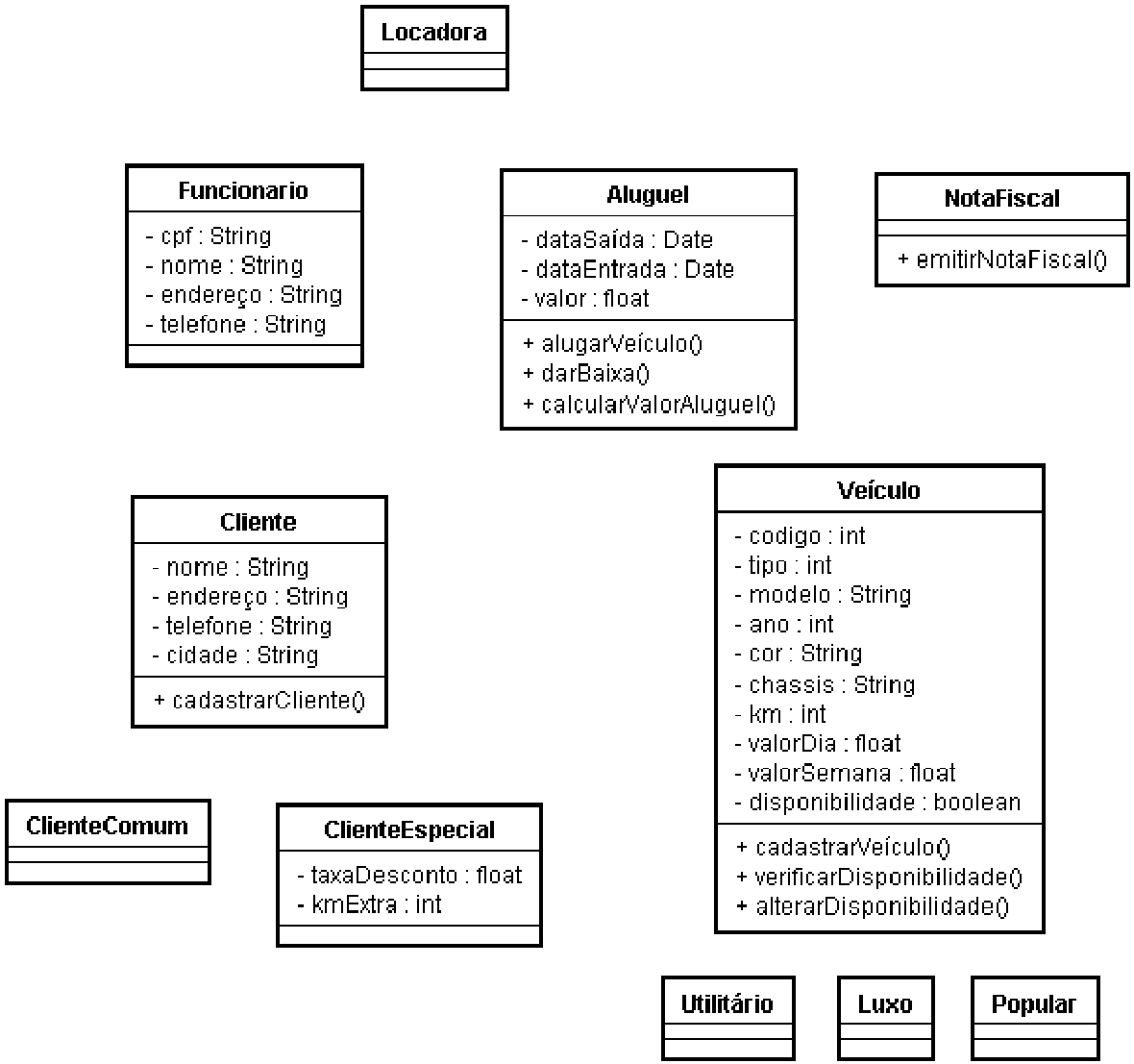
Alocando operações às classes

- Classe deve ter alta coesão – ter apenas operações que sejam de sua natureza.
- Não queremos que carro efetue cadastro de cliente.
- Veículo deve ter a operação cadastrar veículo
- Cliente deve ter a operação cadastrar cliente.
- Estamos modelando um diagrama de classes a nível de domínio. Quando estivermos modelando um diagrama de classe mais próximo da implementação, teremos uma classe controler que vai abrigar essas operações para gerenciar as diferentes entidades que serão persistidas na aplicação.

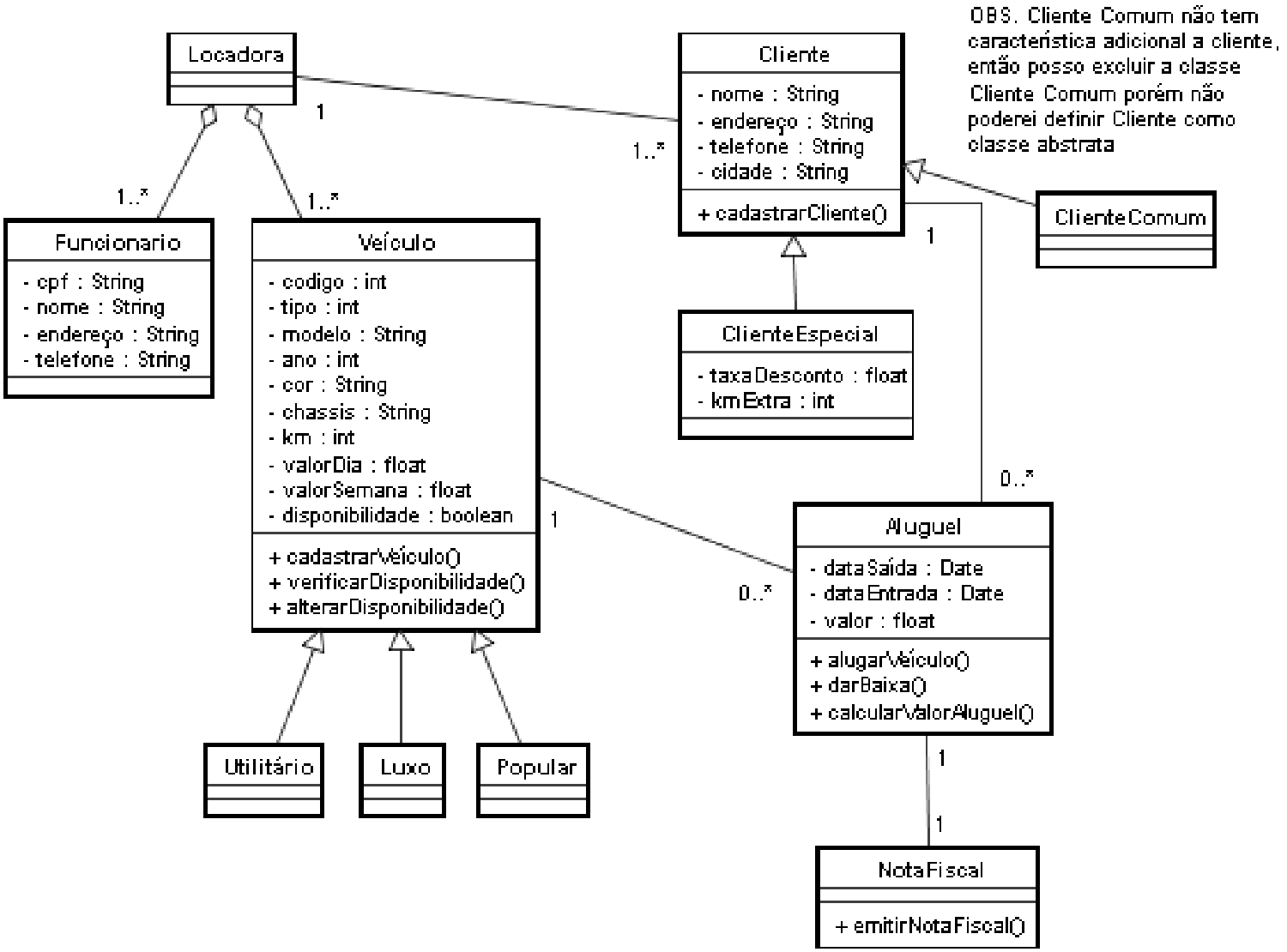
Alocando operações às classes

- alugar veículo – pertence à Aluguel
- cadastrar cliente - pertence à Cliente
- cadastrar veículo - pertence à Veículo
- dar baixa - pertence à Aluguel
- verificar disponibilidade - pertence à Veículo
- alterar disponibilidade - pertence à Veículo
- calcular valor aluguel - pertence à Aluguel
- emitir nota fiscal - pertence à NotaFiscal se quiser manter histórico de notas emitidas se não for gerenciar as notas, poderíamos ter essa operação em Aluguel que possui todos os dados que estarão na nota. Como vamos manter essa classe NotaFiscal, vamos deixar essa operação na classe Notafiscal.

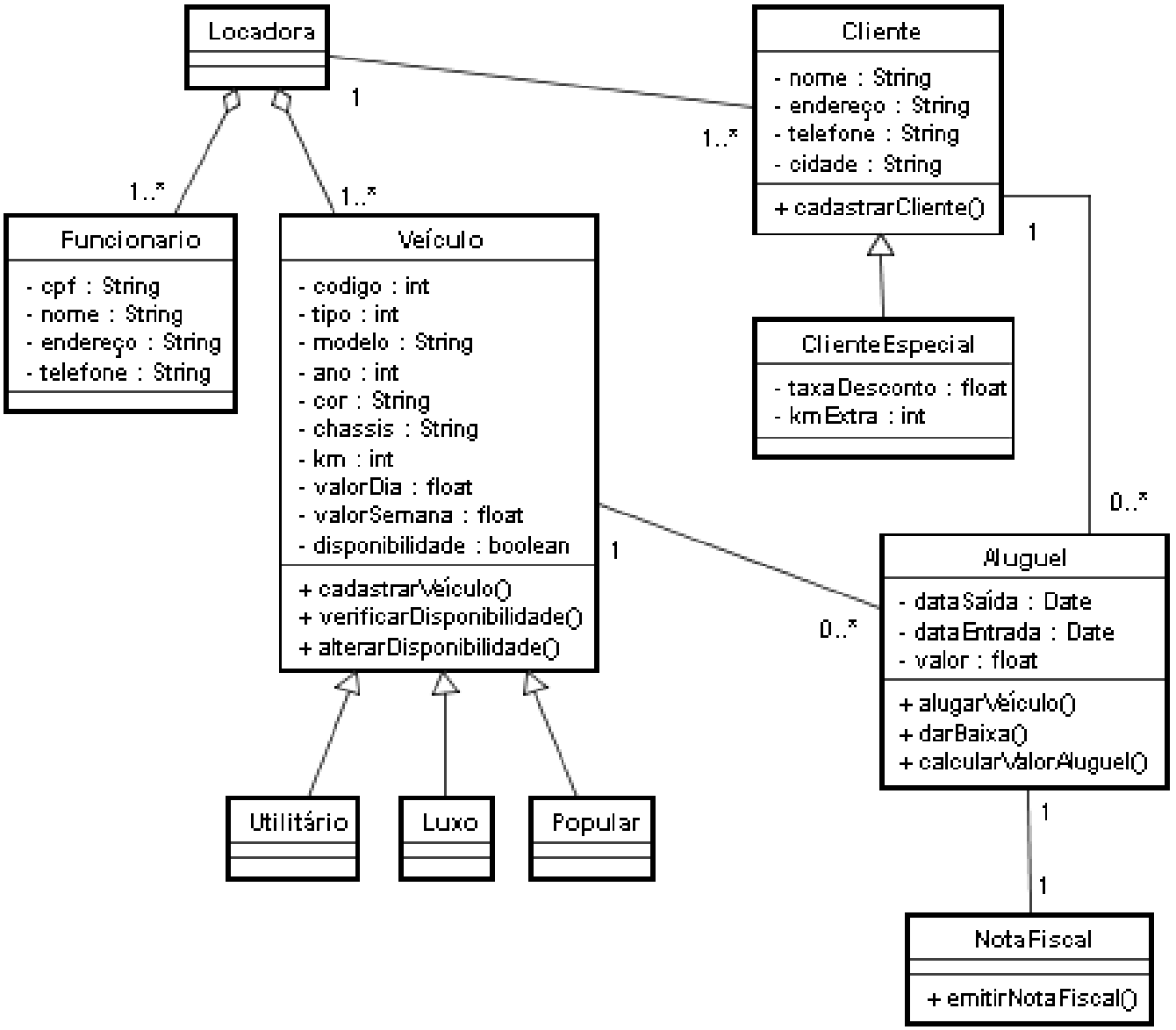
Modelando operações



Criando Associações



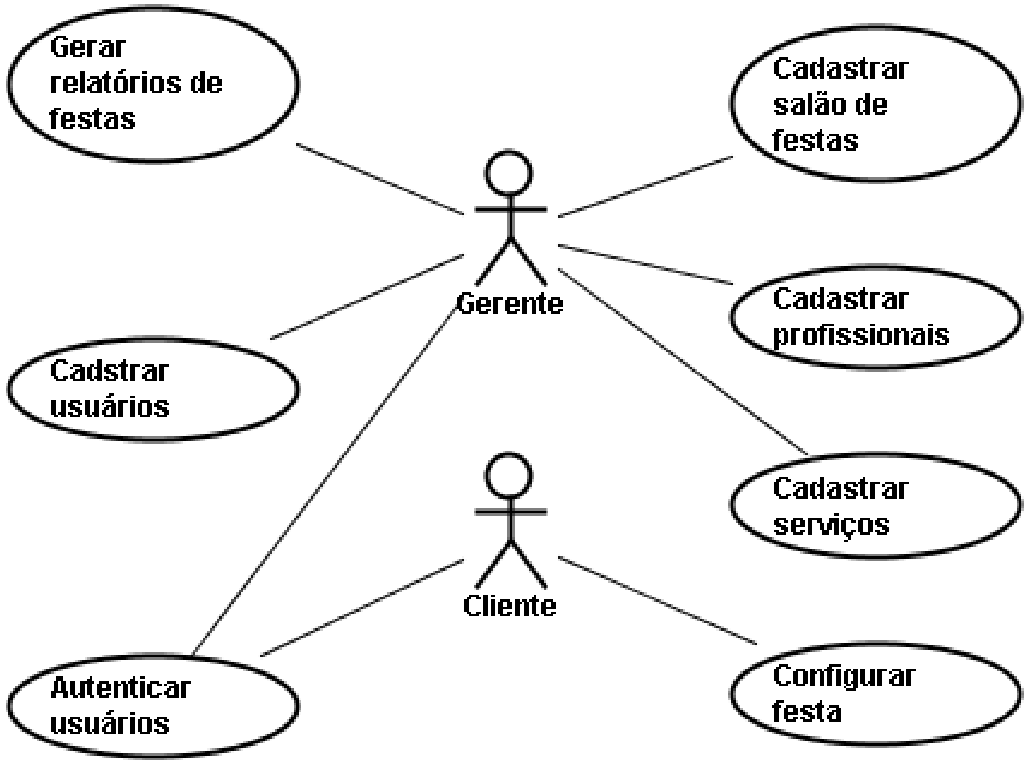
Criando Associações



Exercícios

RF1 O software deverá permitir que o gerente cadastre salões de festas
RF2 O software deverá permitir que o gerente cadastre profissionais
RF3 O software deverá permitir que o gerente realize o cadastramento de serviços utilizados em uma festa de casamento, como buffet, carros utilizados
RF4 O software deverá permitir que o cliente realize o cadastramento da lista de convidados
RF5 O software deverá permitir que o cliente realize o cadastramento das lojas de presentes
RF6 O software deverá permitir que o cliente elabore a configuração da festa
RF7 O software deverá permitir a geração do orçamento para o cliente
RF8 O software deverá permitir que o gerente realize a impressão de relatórios de festas
RF9 O software deverá permitir que o cliente cadastre a lista de presentes
RF10 O software deverá permitir que o gerente realize o cadastramento de usuários
RF11 O software deverá permitir a autenticação dos usuários

Diagrama de caso de uso



Descrição do caso de uso

Caso de Uso: Autenticar Usuário

Objetivo: Permitir que os usuários do sistema sejam autenticados antes de terem acesso às funcionalidades do sistema.

Requisitos: RF11

Atores: Gerente, Cliente

Condição de Início: O usuário acessa o sistema.

Descrição do caso de uso

Fluxo Principal:

1. O sistema apresenta tela de autenticação contendo as informações:
   - Login
   - Senha
   - As opções:
     * Confirmar
     * Cancelar
2. O ator informa login, senha e seleciona a opção Confirmar. [A1]
3. O sistema verifica informações de login. [A2]
4. O sistema autentica o usuário.
5. O sistema disponibiliza as funcionalidades que o ator tem acesso.
6. O caso de uso é encerrado.

Descrição do caso de uso

Fluxo Alternativo: [A1] O ator seleciona a opção Cancelar

1. O sistema retorna para a tela inicial.
2. O caso de uso é encerrado.

[A2] Informações de login inválidas

1. O sistema apresenta a mensagem “Login ou senha inválido” e solicita a confirmação do Ator.
2. Ator confirma.
3. O caso retorna ao passo 1 do fluxo principal.

Regras de negócio: -

Descrição do caso de uso

- Caso de Uso: Cadastrar Usuário
- Objetivo: Permitir que o gerente efetue o cadastro de usuários no sistema.
- Requisitos: RF10
- Atores: Gerente
- Condição de Início: O ator seleciona a opção Cadastrar Usuário.

Descrição do caso de uso

Fluxo Principal:

1. O sistema apresenta a tela de cadastro de usuário contendo as informações:

   - CPF, Nome, Endereço, Data de Nascimento

- As opções:
  * Incluir
  * Cancelar

2. O ator informa os dados de cadastro (CPF, nome, endereço, data de nascimento) e seleciona a opção Incluir. [A1]
3. O sistema verifica se o usuário já está cadastro no sistema. [A2]
4. O sistema verifica se as informações para cadastro são válidas. [A3] [RN1]
5. O sistema confirma o cadastro do usuário.
6. O caso de uso é encerrado.

Descrição do caso de uso

Fluxo Alternativo: [A1] O ator selecionou a opção cancelar

1. O sistema retorna para a tela inicial.
2. O caso de uso é encerrado.

[A2] Usuário já cadastrado

1. O sistema informa que o usuário já se encontra cadastrado.
2. O caso de uso retorna ao passo 1 do fluxo principal.

[A3] Informações de cadastro inválidas

1. O sistema informa que existem campos não preenchidos no cadastro.
2. O caso de uso retorna ao passo 2 do fluxo principal.

Regras de negócio:

[RN1] Todas as informações de cadastro são obrigatórias.

Descrição do caso de uso

- Caso de Uso: Configurar Festa
- Objetivo: Permitir que o cliente efetue a configuração da festa informando quais
- serviços serão utilizados, a lista de convidados e a lista de presentes.
- Requisitos: RF4, RF5, RF6, RF7, RF9
- Atores: Cliente
- Condição de Início: O ator seleciona a opção Configurar Festa.

Descrição do caso de uso

Fluxo Principal: 1. O sistema apresenta tela para definição de serviços para a festa contendo as informações:

(Novo Serviço)

- Serviços Disponíveis (contendo uma lista dos serviços cadastrados no sistema)
- A opção Incluir Serviço

(Serviços Incluídos)

- Serviços Incluídos contendo uma lista dos serviços incluídos com as seguintes informações:
  * Nome do Serviço
  * Opção de Exclusão

2. O sistema apresenta ao final as opções:

- Avançar
- Cancelar

Descrição do caso de uso

3. O ator seleciona um serviço e a opção incluir serviço [A1]
4. O ator seleciona a opção Avançar [A2]
5. O sistema apresenta tela para definição da lista de convidados para a festa contendo as informações:

(Novo Convidado)

- Nome do Convidado e e-mail
- A opção Incluir Convidado

(Convidados Incluídos)

- Convidados Incluídos contendo uma lista dos convidados incluídos com as seguintes informações:
  * Nome do Convidado, e-mail
  * Opção de Exclusão

6. O sistema apresenta ao final as opções:

- Avançar
- Voltar

Descrição do caso de uso

7. O ator informa um convidado e seleciona a opção incluir convidado [A3]
8. O ator seleciona a opção Avançar [A4]
9. O sistema apresenta tela para definição da lista de presentes contendo as informações:

(Novo Presente)

- Nome do Presente
- A opção Incluir Presente

(Presentes Incluídos)

- Presentes Incluídos contendo uma lista dos presentes incluídos com as

seguintes informações:

  * Nome do Presente
  * Opção de Exclusão

Descrição do caso de uso

10. O sistema apresenta ao final as opções:

- Avançar
- Voltar

11. O ator informa um presente e seleciona a opção incluir presente [A5]

12. O ator seleciona a opção Avançar [A6]

13. O sistema apresente a tela de orçamento contendo as seguintes informações:

(Orçamento da Festa)

- Lista contendo para cada serviço definido para a festa as informações:
  * Nome do Serviço * Valor
  - Quantidade de Convidados - Valor (considerando a quantidade de convidados) - Valor Total [RN1]
  - As opções:
    * Confirmar
    * Cancelar

Descrição do caso de uso

14. O ator seleciona a opção Confirmar [A7]
15. O sistema armazena as configurações da festa.
16. O caso de uso é encerrado.

Fluxo Alternativo: [A1] O ator seleciona a opção de exclusão de serviço

1. O sistema retira o serviço da lista de serviços incluídos
2. O caso retorna ao passo 1 do fluxo principal.

[A2] O ator seleciona a opção cancelar

1. O sistema retorna para a tela inicial
2. O caso é encerrado.

Descrição do caso de uso

[A3] O ator seleciona a opção de exclusão de convidado

1. O sistema retira o convidado da lista de convidados incluídos
2. O caso retorna ao passo 5 do fluxo principal.

[A4] O ator seleciona a opção voltar

1. O caso de uso retorna ao passo 1 do fluxo principal.

[A5] O ator seleciona a opção de exclusão de presente

1. O sistema retira o presente da lista de presentes incluídos
2. O caso retorna ao passo 9 do fluxo principal.

[A6] O ator seleciona a opção voltar

1. O caso de uso retorna ao passo 5 do fluxo principal.

[A7] O ator seleciona a opção Cancelar

1. As configurações da festa são desconsideradas.
2. O sistema retorna para a tela inicial
3. O caso de uso é encerrado.

Descrição do caso de uso

Regras de negócio:

[RN1] O valor total é dado pela soma dos serviços incluídos mais o valor definido a partir da quantidade de convidados para festa.

Identificando Classes ou atributos nos RF

- Gerente
- Salão de festas
- Profissional
- Serviço
- Festa de Casamento
- Buffet
- Carro
- Cliente
- Lista de Convidado
- Loja de Presente
- Configuração da Festa
- Orçamento
- Relatório da festa
- Lista de Presente
- Usuário

Identificando Classes ou atributos nos casos de uso

- Gerente
- Salão de festas
- Profissional
- Serviço
- Festa de Casamento
- Buffet
- Carro
- Cliente
- Lista de Convidado
- Loja de Presente
- Configuração da Festa
- Orçamento
- Relatório da festa
- Lista de Presente
- Usuário

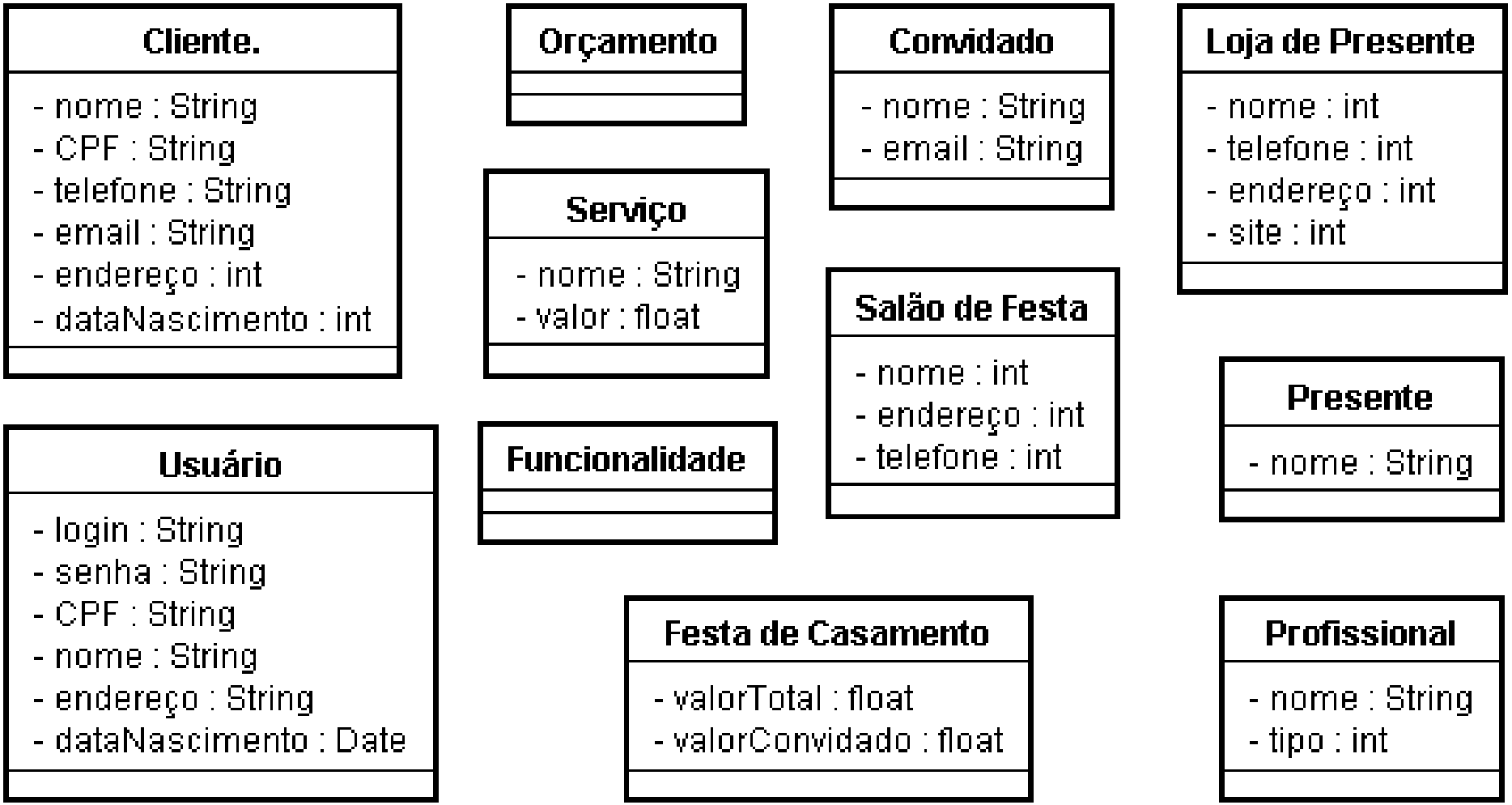
- Login
- Senha
- Funcionalidade do ator (o que o ator terá acesso)
- CPF
- Nome
- Endereço
- Data nascimento
- Nome do Serviço
- Nome do Convidado
- E-mail convidado
- Nome do presente
- Valor do serviço
- Quantidade de convidados
- Valor
- Valor Total da Festa

Diferenciando Classes de Atributos

- Gerente (tipo de usuário - ator)
- Salão de festas
- Profissional
- Serviço
- Festa de Casamento
- Buffet (instância de serviço)
- Carro (instância de serviço)
- Cliente
- Lista de Convidado
- Loja de Presente
- Configuração da Festa (sinônimo de festa de casamento)
- Orçamento ?
- Relatório da festa (emitir relatório –ação)
- Lista de Presente
- Usuário

- Login (Usuário)
- Senha (Usuário)
- Funcionalidade do ator (o que o ator terá acesso)
- CPF (Usuário)
- Nome (Usuário)
- Endereço (Usuário)
- Data nascimento (Usuário)
- Nome do Serviço (Serviço)
- Nome do Convidado (Convidado)
- E-mail (Convidado)
- Nome do presente (Presente)
- Valor do serviço(Serviço)
- Quantidade de convidados (instância – derivado)
- Valor por convidado (Festa)
- Valor Total da Festa (Festa de casamento)

Modelando Classes



Identificando comportamentos nos RF

RF1 O software deverá permitir que o gerente cadastre salões de festas
RF2 O software deverá permitir que o gerente cadastre profissionais
RF3 O software deverá permitir que o gerente realize o cadastramento de serviços utilizados em uma festa de casamento, como buffet, carros utilizados
RF4 O software deverá permitir que o cliente realize o cadastramento da lista de convidados
RF5 O software deverá permitir que o cliente realize o cadastramento das lojas de presentes
RF6 O software deverá permitir que o cliente elabore a configuração da festa
RF7 O software deverá permitir a geração do orçamento para o cliente
RF8 O software deverá permitir que o gerente realize a impressão de relatórios de festas
RF9 O software deverá permitir que o cliente cadastre a lista de presentes
RF10 O software deverá permitir que o gerente realize o cadastramento de usuários
RF11 O software deverá permitir a autenticação dos usuários

Lista de comportamentos encontrados

cadastrar salão de festas (consulta, inclusão, exclusão e alteração)
cadastrar profissionais (consulta, inclusão, exclusão e alteração)
cadastrar serviços (consulta, inclusão, exclusão e alteração)
cadastrar lista de convidados (Para o diagrama de classe a nível de domínio, vamos encarar essa funcionalidade como cadastrar convidado)
cadastrar lojas de presentes (consulta, inclusão, exclusão e alteração)
configurar festa
gerar orçamento
imprimir relatório de festas
cadastrar a lista de presentes (Para o diagrama de classe a nível de domínio, vamos encarar essa funcionalidade como cadastrar presente)
cadastrar usuários (consulta, inclusão, exclusão e alteração)
autenticar usuários

Identificando comportamentos no caso de Uso - Autenticar Usuário

Fluxo Principal:

1. O sistema apresenta tela de autenticação contendo as informações:
   - Login
   - Senha
   - As opções:
     * Confirmar
     * Cancelar
2. O ator informa login, senha e seleciona a opção Confirmar. [A1]
3. O sistema verifica informações de login. [A2]
4. O sistema autentica o usuário. (já identificado nos RF)
5. O sistema disponibiliza as funcionalidades que o ator tem acesso.
6. O caso de uso é encerrado.

Não procuramos interações com interface e base de dados, e sim do contexto do sistema

Lista de comportamentos encontrados

cadastrar salão de festas (consulta, inclusão, exclusão e alteração)
cadastrar profissionais (consulta, inclusão, exclusão e alteração)
cadastrar serviços (consulta, inclusão, exclusão e alteração)
cadastrar convidado (consulta, inclusão, exclusão e alteração)
cadastrar lojas de presentes (consulta, inclusão, exclusão e alteração)
configurar festa
gerar orçamento
imprimir relatório de festas
cadastrar presente (consulta, inclusão, exclusão e alteração)
cadastrar usuários (consulta, inclusão, exclusão e alteração)
autenticar usuários
    verificar validade das informações de login
    disponibilizar as funcionalidades do sistema

Identificando comportamentos no caso de Uso - Autenticar Usuário

Fluxo Alternativo: [A1] O ator seleciona a opção Cancelar

1. O sistema retorna para a tela inicial.
2. O caso de uso é encerrado.

[A2] Informações de login inválidas

1. O sistema apresenta a mensagem “Login ou senha inválido” e solicita a confirmação do Ator.
2. Ator confirma.
3. O caso retorna ao passo 1 do fluxo principal.

Regras de negócio: -

Não temos interações adicionais do contexto do sistema aqui, vamos em frente

Identificando comportamentos no Caso de Uso - Cadastrar Usuário

Fluxo Principal:

1. O sistema apresenta a tela de cadastro de usuário contendo as informações:

   - CPF, Nome, Endereço, Data de Nascimento

- As opções:
  * Incluir
  * Cancelar

2. O ator informa os dados de cadastro (CPF, nome, endereço, data de nascimento) e seleciona a opção Incluir. [A1]
3. O sistema verifica se o usuário já está cadastrado no sistema. [A2]
4. O sistema verifica se as informações para cadastro são válidas. [A3] [RN1]
5. O sistema confirma o cadastro do usuário.
6. O caso de uso é encerrado.

Lista de comportamentos encontrados

cadastrar salão de festas (consulta, inclusão, exclusão e alteração)
cadastrar profissionais (consulta, inclusão, exclusão e alteração)
cadastrar serviços (consulta, inclusão, exclusão e alteração)
cadastrar convidado (consulta, inclusão, exclusão e alteração)
cadastrar lojas de presentes (consulta, inclusão, exclusão e alteração)
configurar festa
gerar orçamento
imprimir relatório de festas
cadastrar presente (consulta, inclusão, exclusão e alteração)
cadastrar usuários (consulta, inclusão, exclusão e alteração)
    verificar se usuário já é cadastrado
    verificar validade das informações de cadastro
autenticar usuários
    verificar validade das informações de login
    disponibilizar as funcionalidades do sistema

Identificando comportamentos no Caso de Uso - Cadastrar Usuário

Fluxo Alternativo: [A1] O ator selecionou a opção cancelar

1. O sistema retorna para a tela inicial.
2. O caso de uso é encerrado.

[A2] Usuário já cadastrado

1. O sistema informa que o usuário já se encontra cadastrado.
2. O caso de uso retorna ao passo 1 do fluxo principal.

[A3] Informações de cadastro inválidas

1. O sistema informa que existem campos não preenchidos no cadastro.
2. O caso de uso retorna ao passo 2 do fluxo principal.

Regras de negócio:

[RN1] Todas as informações de cadastro são obrigatórias.

Não temos interações adicionais do contexto do sistema aqui, vamos em frente

Identificando comportamentos no Caso de Uso - Configurar Festa

Fluxo Principal: 1. O sistema apresenta tela para definição de serviços para a festa contendo as informações:

(Novo Serviço)

- Serviços Disponíveis (contendo uma lista dos serviços cadastrados no sistema)
- A opção Incluir Serviço

(Serviços Incluídos)

- Serviços Incluídos contendo uma lista dos serviços incluídos com as seguintes informações:
  * Nome do Serviço
  * Opção de Exclusão

2. O sistema apresenta ao final as opções:

- Avançar
- Cancelar

Assim devemos ter uma funcionalidade para consultar esses serviços e apresentá-los

Assim devemos ter uma funcionalidade para recuperar os serviços da festa em questão e apresentá-los. Obs: a assinatura desses dois métodos são diferentes...

Identificando comportamentos no Caso de Uso - Configurar Festa

3. O ator seleciona um serviço e a opção incluir serviço [A1]
4. O ator seleciona a opção Avançar [A2]
5. O sistema apresenta tela para definição da lista de convidados para a festa contendo as informações:

(Novo Convidado)

- Nome do Convidado e e-mail
- A opção Incluir Convidado

(Convidados Incluídos)

- Convidados Incluídos contendo uma lista dos convidados incluídos com as seguintes informações:
  * Nome do Convidado, e-mail
  * Opção de Exclusão

6. O sistema apresenta ao final as opções:

- Avançar ou - Voltar

Identificando comportamentos no Caso de Uso - Configurar Festa

7. O ator informa um convidado e seleciona a opção incluir convidado [A3]
8. O ator seleciona a opção Avançar [A4]
9. O sistema apresenta tela para definição da lista de presentes contendo as informações:

(Novo Presente)

- Nome do Presente
- A opção Incluir Presente

(Presentes Incluídos)

- Presentes Incluídos contendo uma lista dos presentes incluídos com as seguintes informações:
  * Nome do Presente
  * Opção de Exclusão

Identificando comportamentos no Caso de Uso - Configurar Festa

10. O sistema apresenta ao final as opções:

- Avançar - Voltar

11. O ator informa um presente e seleciona a opção incluir presente [A5]

12. O ator seleciona a opção Avançar [A6]

13. O sistema apresente a tela de orçamento contendo as seguintes informações:

(Orçamento da Festa)

- Lista contendo para cada serviço definido para a festa as informações:
  * Nome do Serviço * Valor
  - Quantidade de Convidados
- Valor (considerando a quantidade de convidados)
- Valor Total [RN1]
  - As opções:
    * Confirmar * Cancelar

Esses dois valores serão calculados pelo sistema, assim devemos ter as funcionalidades que serão responsáveis por estes cálculos

Identificando comportamentos no Caso de Uso - Configurar Festa

14. O ator seleciona a opção Confirmar [A7]
15. O sistema armazena as configurações da festa.
16. O caso de uso é encerrado.

Fluxo Alternativo: [A1] O ator seleciona a opção de exclusão de serviço

1. O sistema retira o serviço da lista de serviços incluídos
2. O caso retorna ao passo 1 do fluxo principal.

[A2] O ator seleciona a opção cancelar

1. O sistema retorna para a tela inicial
2. O caso é encerrado.

Identificando comportamentos no Caso de Uso - Configurar Festa

[A3] O ator seleciona a opção de exclusão de convidado

1. O sistema retira o convidado da lista de convidados incluídos
2. O caso retorna ao passo 5 do fluxo principal.

[A4] O ator seleciona a opção voltar

1. O caso de uso retorna ao passo 1 do fluxo principal.

[A5] O ator seleciona a opção de exclusão de presente

1. O sistema retira o presente da lista de presentes incluídos
2. O caso retorna ao passo 9 do fluxo principal.

[A6] O ator seleciona a opção voltar

1. O caso de uso retorna ao passo 5 do fluxo principal.

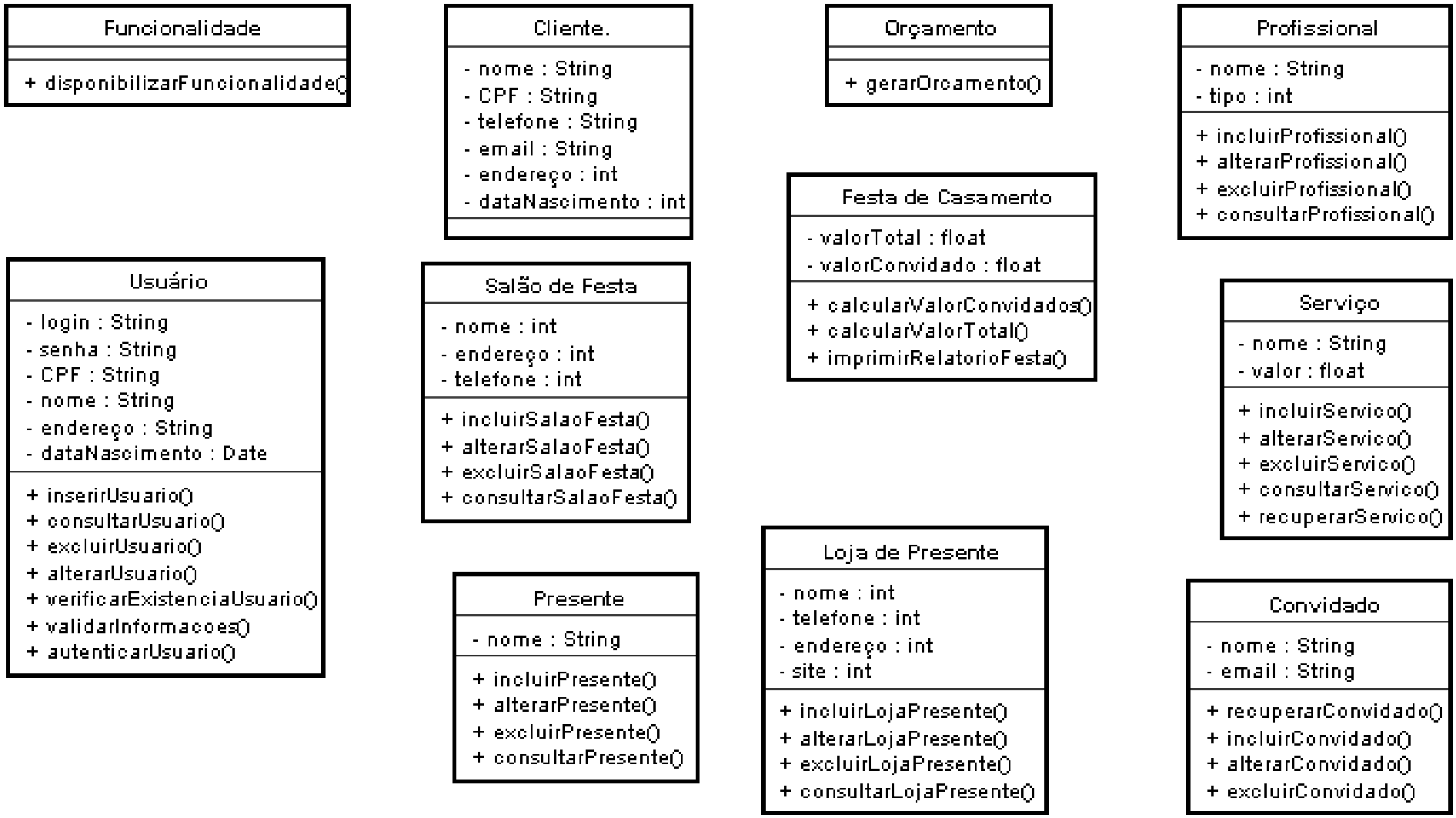
[A7] O ator seleciona a opção Cancelar

1. As configurações da festa são desconsideradas.
2. O sistema retorna para a tela inicial
3. O caso de uso é encerrado.

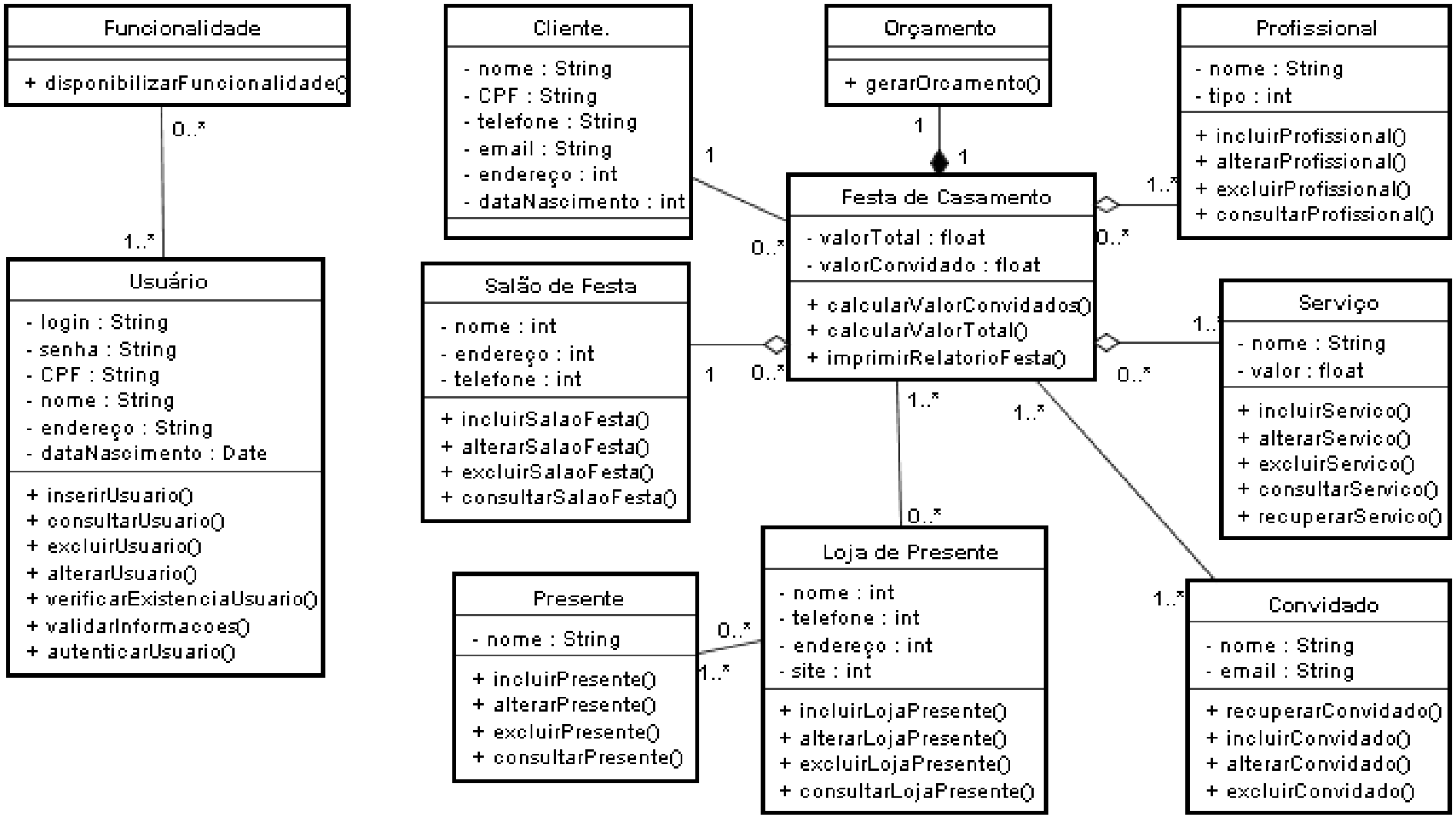
Lista de comportamentos encontrados

cadastrar salão de festas (consulta, inclusão, exclusão e alteração)
cadastrar profissionais (consulta, inclusão, exclusão e alteração)
cadastrar serviços (consulta, inclusão, exclusão e alteração)
cadastrar convidado (consulta, inclusão, exclusão e alteração)
cadastrar lojas de presentes
configurar festa
recuperar serviços (geral todos os cadastrados)
consultar serviços para festa (que está associado a festa em específico)
cadastrar presente (consulta, inclusão, exclusão e alteração)
calcular valor de convidados
calcular valor total da festa
gerar orçamento
imprimir relatório de festas
cadastrar usuários (consulta, inclusão, exclusão e alteração)
verificar se usuário já é cadastrado
verificar validade das informações de cadastro
autenticar usuários
verificar validade das informações de login
disponibilizar as funcionalidades do sistema

Modelando Comportamentos



Identificando associações



Exercício
Diagrama de classes

Sistema de controle de utilização de
serviços de uma
operadora de telefonia celular.

O sistema deve contemplar as seguintes regras de negócio:

(a) Os clientes da empresa utilizam os serviços de telefonia celular e pagam por eles, de acordo com o uso;

(b) É necessário armazenar alguns dados dos clientes, tais como: cpf, nome, endereço e data de nascimento;

(c) Os clientes aderem a um plano da operadora. Esse plano pode ser pré-pago ou pós-pago. O valor das tarifas de cada serviço é diferenciada, de acordo com o plano adotado;

(d) O sistema deve possibilitar a compra de cartões pré-pagos;

(e) O sistema deve possibilitar dois tipos de habilitação (pré-pago e pós pago). Ao habilitar na modalidade “pré-pago”, o cliente deve obrigatoriamente comprar um cartão pré-pago;

(f) Os serviços oferecidos pela operadora são: ligações telefônicas, secretária eletrônica, conferência e mensagem de texto;

(g) Caso um cliente atrase o pagamento de sua conta por mais de 60 dias, todos os seus serviços contratados devem ser bloqueados;

(h) Se o cliente efetuar os pagamentos atrasados, os seus serviços devem ser desbloqueados;

(i) A operadora proporciona três formas de pagamento: emissão de boleto bancário, lançamento na fatura do cartão de crédito, ou pagamento através da compra de cartões pré-pagos.

Exercício
Diagrama de classes

Sistema de controle de empréstimo e a devolução de exemplares de uma biblioteca.

Exercício

- Queremos construir um sistema de software para controlar o empréstimo e a devolução de exemplares de uma biblioteca. O usuário pode fazer um empréstimo de um exemplar durante um certo período e, ao final desse tempo, o exemplar deve ser devolvido. Renovações não são aceitas.

- A atendente é uma funcionária que interage com os usuários e com o sistema de controle da biblioteca através de um terminal. As principais características do sistema são listadas a seguir:

- 1.Um usuário do sistema, que pode ser um aluno, um professor ou um outro funcionário da universidade, pode reservar publicações e também cancelar reservas previamente agendadas.

- 2.Um usuário deve estar devidamente cadastrado no sistema para usar os seus serviços. O sistema é operado pela atendente da biblioteca, que também é uma funcionária da universidade.

- 3.Um usuário pode emprestar exemplares previamente reservados ou não. Se foi feita uma reserva, ela deve ser cancelada no momento do seu empréstimo.

- 4.No caso da devolução de um exemplar em atraso, existe uma multa que deve ser paga. Essa multa é calculada com base no número de dias em atraso. Além disso, se o exemplar estiver atrasado por mais de 30 dias e se o usuário não for um professor, além de pagar a multa, o usuário ´e suspenso por um período de 2 meses.

- 5.Um exemplar da biblioteca pode ser bloqueado/desbloqueado por um professor por um período de tempo. Nesse caso, o exemplar fica disponível numa estante, podendo ser consultado por usuários da biblioteca, mas não pode ser emprestado.

- 6.O período de empréstimo é variável, dependendo do tipo de usuário (7 dias para alunos e funcionário, e 15 dias para professores).

- 7.A manutenção dos dados do acervo da biblioteca é feita pela bibliotecária, que também é funcionária da universidade. Ela é responsável pelo inclusão de novos exemplares, exclusão de exemplares antigos e pela atualização dos dados dos exemplares cadastrados.

- Os exemplares podem ser livros, periódicos, manuais e teses. As publicações são identificadas pelo seu número do tombo, além de outras características como o título, nome do autor, editora e número da edição correspondente.

- A biblioteca só empresta suas obras para usuários cadastrados. Um usuário é identificado através de seu número de registro. Outras informações relevantes são seu nome, instituto/faculdade a que pertence e seu tipo (aluno/funcionário/professor).

Caminhando um pouco mais...

- Os diagramas de classe que vimos é o diagrama de classe de análise - está modelado a nível de domínio.
- A etapa de análise foca no entendimento dos requisitos, conceitos e operações relacionados com o sistema. Ela enfatiza o que deve ser feito. Essa etapa foca no domínio do problema (ambiente)
- Depois da Analise, vem o Projeto e daí o foca muda para o domínio da solução.
- Assim, a etapa de projeto produz um diagrama de classes de projeto, similar ao diagrama de classe de análise, só que mais detalhado e focado no domínio da solução.
- No projeto detalhado, as classes de projeto são especificadas de maneira minuciosa; atributos e estruturas de dados que devem ser mantidos em tempo de execução são definidos e operações são descritas de maneira precisa.

Três Camadas

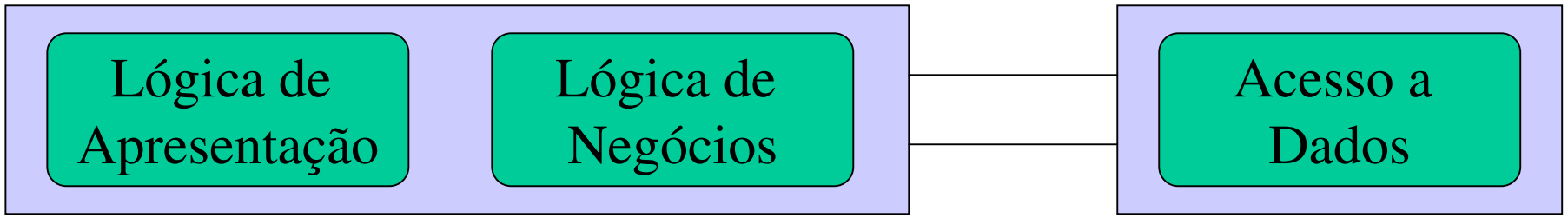
- A organização em camadas é a chave para a independência entre os componentes e esta independência é que vai atingir os objetivos de eficiência , escalabilidade , reutilização e facilidade de manutenção.
- Num primeiro instante produzir aplicativos multicamadas pode parecer mais complexo por isto vou procurar mostrar como evoluímos para chegar a esta solução. Na produção de software vamos considerar camada como uma referência a separação de responsabilidades.

Aplicações monolíticas

- Antigamente um aplicativo era desenvolvido para ser usado em uma única máquina . Geralmente este aplicativo continha todas a funcionalidades em um único módulo gerado por uma grande quantidade de linhas de código e de manutenção difícil. A entrada do usuário , verificação , lógica de negócio e acesso a banco de dados estava presente em um mesmo lugar. Podemos definir este tipo de aplicação como aplicação de uma camada ou monolítica.

Aplicações em duas camadas

- Surgiu pela necessidade de compartilhar a lógica de acesso a dados entre vários usuários simultâneos. Nesta estrutura a base de dados foi colocada em uma máquina específica, separada das máquinas que executavam as aplicações. Temos aplicativos instalados em estações clientes contendo toda a lógica da aplicação (cliente rico ou gordo). Um grande problema neste modelo é o gerenciamento de versões pois para cada alteração os aplicativos precisam ser atualizados em todas as máquinas clientes.



Aplicações em três camadas

- A internet trouxe um movimento para separar a lógica de negócio da interface com o usuário. A idéia é que os usuários da WEB possam acessar as mesmas aplicações sem ter que instalar estas aplicações em suas máquinas locais. Como a lógica do aplicativo , inicialmente contida no cliente não reside mais na máquina do usuário este tipo de cliente passa a ser chamado de cliente pobre ou magro.(thin).
- Neste modelo o aplicativo é movido para o Servidor e um navegador Web é usado como um cliente magro. O aplicativo é executado em servidores Web com os quais o navegador Web se comunica e gera o código HTML para ser exibido no cliente.

Aplicações em três camadas

- A separação em camadas lógicas torna os sistemas mais flexíveis permitindo que as partes possam ser alteradas de forma independente. As funcionalidades da camada de negócio podem ser divididas em classes e essas classes podem ser agrupadas em pacotes ou componentes reduzindo as dependências entre as classes e pacotes; podem ser reutilizadas por diferentes partes do aplicativo e até por aplicativos diferentes. O modelo de 3 camadas tornou-se a arquitetura padrão para sistemas corporativos com base na Web.
- A modelagem orientada a objetos ajuda a promover a modularidade pois os objetos encapsulam seus dados (propriedades, estados) e oferecem funcionalidades através de seus métodos. Projetando-se de forma adequada os objetos podem ter reduzidas as dependências entre si ficando assim fracamente acoplados e serão mais fáceis de manter e evoluir.

O Padrão MVC

- MVC - (Modelo Visualização Controle)

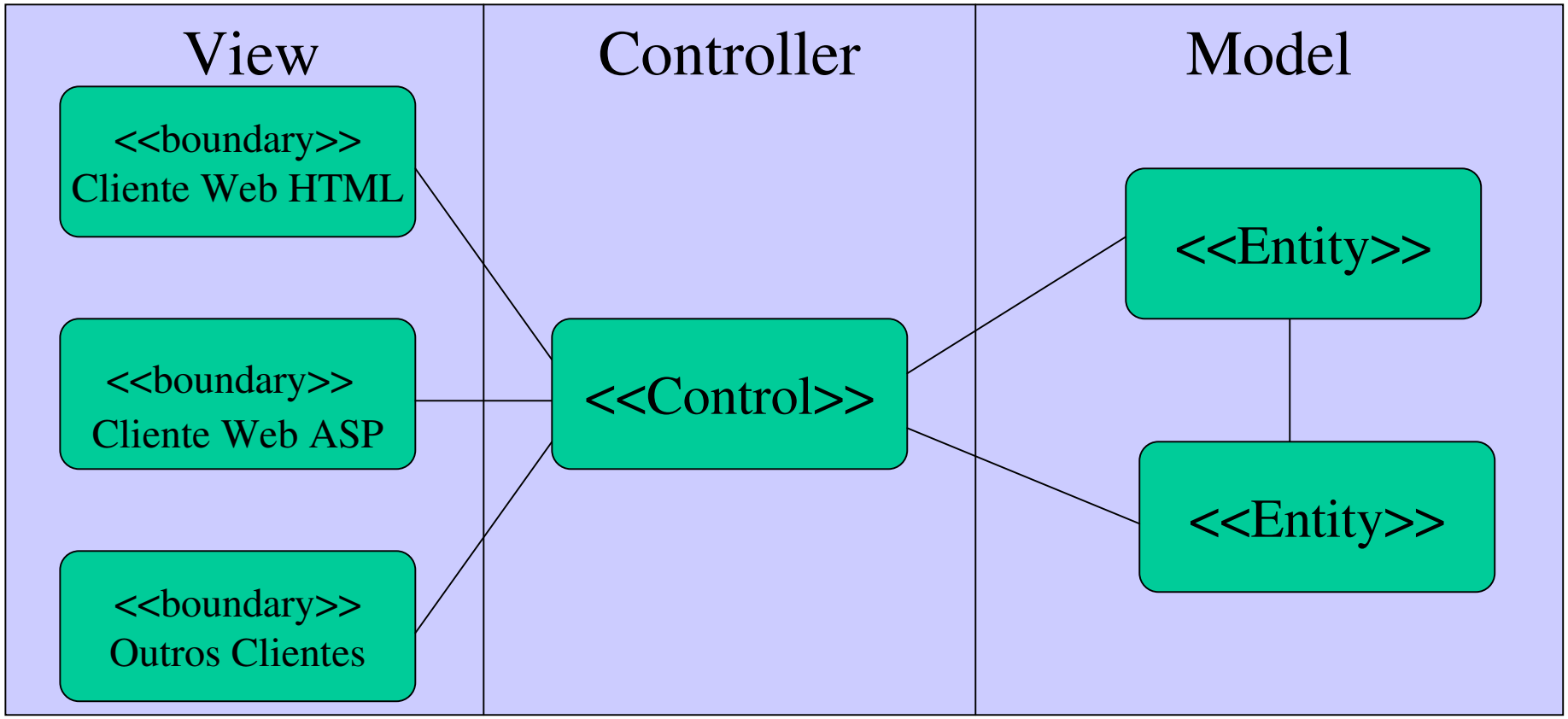
- Esse *padrão*, é muito importante por definir rigorosamente as responsabilidades de cada módulo do projeto, possibilitando ao programador, ou equipe de desenvolvimento, substituir um desses módulos sem interferir nos outros, ou seja, modificar o modo como o objeto é persistido, sem ser obrigado a alterar uma linha de código na classe que o valida, ou então na classe visual que é usada para consultar ou cadastrar esse objeto, por exemplo.

O Padrão MVC

- O MVC define três camadas para a aplicação, são elas:
  - View – Todas as classes que representam interface com o usuário.
  - Controller – Classes utilizadas para validar regras de negócio, processar os dados, aqui conteria toda a lógica de negócio.
  - Model – Utilizando a nomenclatura adotada pelos *patterns* orientados a objetos, essa camada conterá as classes catálogo (classe que persiste e recupera um tipo específico de objeto), classes de entidade (famosos *beans1*, classes que definem um tipo específico de dados com métodos *gets* e *sets*).

O Padrão MVC

- A arquitetura de 3 camadas que esta representada abaixo é uma implementação do modelo MVC . O modelo MVC esta preocupado em separar a informação de sua apresentação.



Classe Limite <<boundary>>

- Representa as relações entre os atores (mundo externo) e o sistema
- Dependendo do tipo de ator, uma Classe Limite é requerida para representar interfaces com:
  - o usuário (atores humanos),
  - os sistemas externos (sistema legado) ou
  - um dispositivo externo atrelado ao sistema
- Esses objetos traduzem os eventos gerados por um ator em eventos relevantes ao sistema
- Também são responsáveis por apresentar os resultados de uma interação dos objetos em algo inteligível pelo ator

Classe Limite <<boundary>>

- É altamente dependente do ambiente
- Normalmente possui as seguintes responsabilidades:
  - Notificar aos objetos de controle, dados sobre os eventos gerados externamente ao sistema
  - Notificar aos atores o resultado de interações entre os objetos internos
- É o único tipo de classe que pode interagir com o ator do sistema

Classe Controle <<control>>

- Separa as classes de interface das classes da lógica do negócio
- Responsável por controlar a lógica de execução correspondente a um caso de uso
- Decide o que o sistema deve fazer quando um evento externo relevante ocorre
  - Realiza o controle do processamento
  - Age como gerente (coordenadores, controladores) dos outros objetos para a realização de um ou mais casos de uso
- As instâncias da Classe Controle normalmente têm vida curta e existem somente durante a realização de um caso de uso

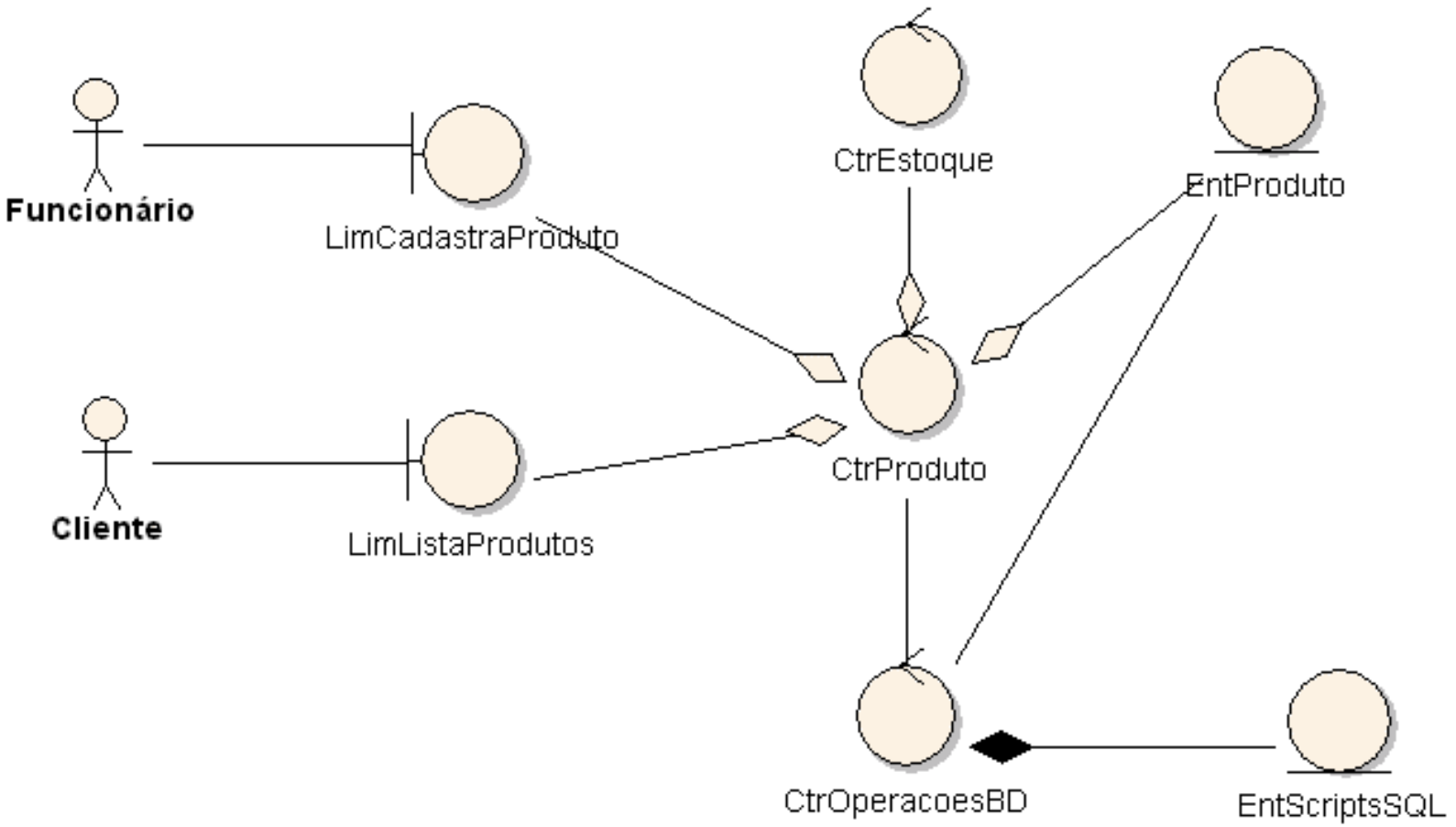
Classe Controle <<control>>

- Responsabilidades típicas
  - Realizar monitorações a fim de responder a eventos externos ao sistema (gerados por objetos limite)
  - Coordenar a realização de um caso de uso através do envio de mensagens a objetos limite e objetos entidade
  - Assegurar que as regras do negócio estejam sendo seguidas corretamente
  - Coordenar a criação de associações entre objetos entidade

Classe Entidade <<entity>>

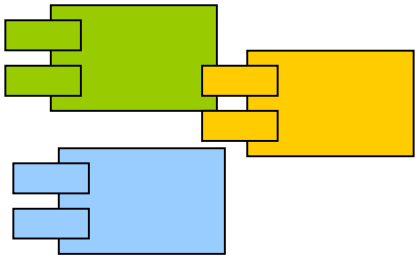
- São classes básicas do domínio do problema
- Atores não têm acesso direto a esses objetos
- Gerencia as informações de que o sistema necessita para fornecer a funcionalidade requerida
- Geralmente as classes da entidade são específicas do negócio
- São informações especializadas e encapsulam o conhecimento do negócio
- Na maioria das vezes, as classes da entidade são as classes persistentes que podem ser usadas para gerar diretamente o esquema da base de dados
  - É um repositório para alguma informação manipulada pelo sistema

Exemplo

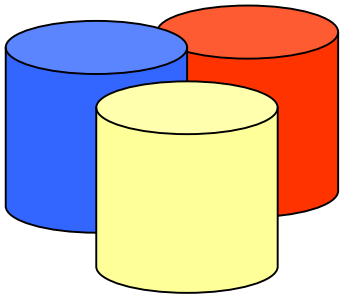

Funcionário
LimCadastraProduto
CtrEstoque
EntProduto
Cliente
LimListaProdutos
CtrProduto
CtrOperacoesBD
EntScriptsSQL

Arquitetura MVC

Interação com usuários ou com outros sistemas



Componentes de negócio



Persistência de dados

View

Controller

Model

Objetivo Arquitetura MVC

- Separar:
  - dados do negócios (Unid. Inf.) (Model);
  - da interface do usuário (View) e;
  - do fluxo e regras da aplicação (Control)
- O modelo do negócio não deve conhecer nada sobre as telas que exibem seu estado.

Exercício

- Refinar o Diagrama de Classes do exercício da biblioteca com MVC