

# Construção de Sistemas de Software

## Transaction Script

### Padrões para Aplicações Empresariais

#### 1. O Sistema SalesSys.

O SalesSys é um sistema empresarial de vendas e administração de inventário. O propósito da aplicação é fazer a gestão da carteira de clientes e de vendas de uma empresa. É um aplicação fictícia, muito simples, que vai servir para exemplificar e exercitar os conceitos e técnicas lecionadas na disciplina. Durante as próximas aulas iremo-nos focar no desenvolvimento da primeira iteração deste sistema, explorando e comparando diferentes soluções de desenho e de implementação, obtidas por aplicação de diferentes padrões para aplicações empresariais.

Para a primeira iteração do sistema, como é habitual, vamos considerar um fragmento simplificado do problema, centrado nas vendas que a empresa realiza aos seus clientes.

A empresa possui uma carteira de clientes com quem comercializa produtos. Nesta primeira fase, cada cliente é caracterizado por um número de pessoa coletiva, uma denominação (nome do cliente) e um contacto telefónico. Cada produto comercializado pela empresa tem um código que o identifica univocamente, uma descrição e um preço. Um produto tem, em cada momento, uma quantidade de artigos disponíveis em stock.

As vendas registam transações efetuadas com os clientes. Cada venda grava a data em que a operação foi efetuada, o cliente, a lista de produtos (e respetivas quantidades) transacionadas e o valor total. A empresa pratica, nas vendas que efetua, variados tipos de descontos. Assim, cada venda regista também o valor do desconto aplicado.

Atualmente a empresa pratica dois tipos de desconto nas suas vendas: o primeiro resulta da aplicação de uma percentagem sobre o valor global, caso o cliente atinja um determinado valor total na compra; o segundo corresponde a uma percentagem sobre o total dos produtos que estiverem marcados como elegíveis para desconto. O tipo de desconto a que

uma venda está sujeita é determinada pelo cliente a quem a venda é efetuada. Um cliente pode não ter direito a nenhum tipo de desconto.

A empresa prevê num futuro próximo enriquecer as formas de desconto com o objetivo de acompanhar a concorrência no setor. Para tal, além de novos modelos de desconto, prevê também poder vir a combinar vários descontos numa mesma venda, mas para já ainda não é um requisito da aplicação.

Os casos de uso definidos para a primeira versão do sistema foram apenas 5: adicionar um novo cliente, criar uma nova venda, adicionar uma entrada a uma venda, obter o valor de desconto de uma venda e fechar uma venda. Em todos os casos foi definido que o caso de uso incluía apenas uma operação que nos seus parâmetros carrega toda a informação necessária.

Foi ainda decidido que a aplicação vai ter uma arquitetura em camadas e ser implementada em Java.



#### (a) Aplicações Empresariais

Enumere um conjunto de características das aplicações empresariais que o SalesSys potencialmente irá possuir, justificando.

#### (b) Arquitetura em Camadas

Explique o que implica em concreto a decisão de que a aplicação deve ter uma arquitetura em camadas. Enumere as camadas que considera que devem ser concretizadas, justificando, e as responsabilidades atribuídas a cada uma delas.

#### (c) Modelo de Dados

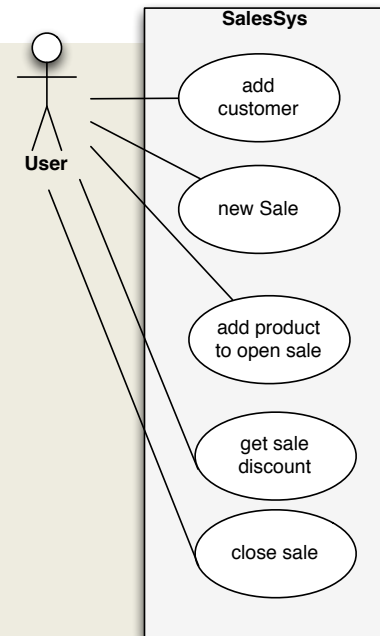
Suponha que foi decidido usar uma base de dados relacional para persistir todos os dados da aplicação SalesSys. Apresente um modelo de dados apropriado.

#### (d) Organização da camada de negócio: Domain Model

Suponha que se pretende desenhar a camada de negócio do SalesSys de acordo com o padrão Domain Model, ou seja, pretende-se

## SalesSys: Casos de Uso

O SalesSys é um sistema empresarial de vendas e administração de inventário. O propósito da aplicação é fazer a gestão da carteira de clientes e de vendas de uma empresa.



## Exercícios (1 b)

Explique o que implica em concreto a decisão de que a aplicação deve ter uma arquitetura em camadas. Enumere as camadas que considera que devem ser concretizadas, justificando, e as responsabilidades de cada uma delas.

## Exercícios (1 b)

---

slide 10 @ 03 CSS ea-arch



## Exercícios (1 c)

---

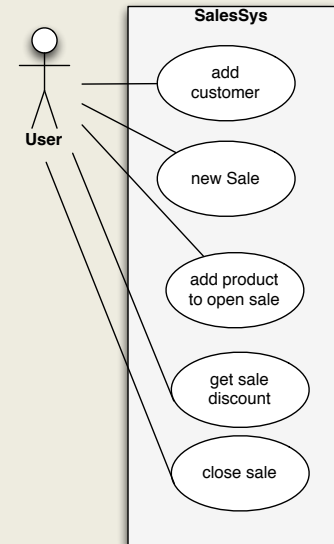
Suponha que foi decidido usar uma base de dados relacional para persistir todos os dados da aplicação. Apresente um modelo de dados apropriado.

## Exercícios (1 c)

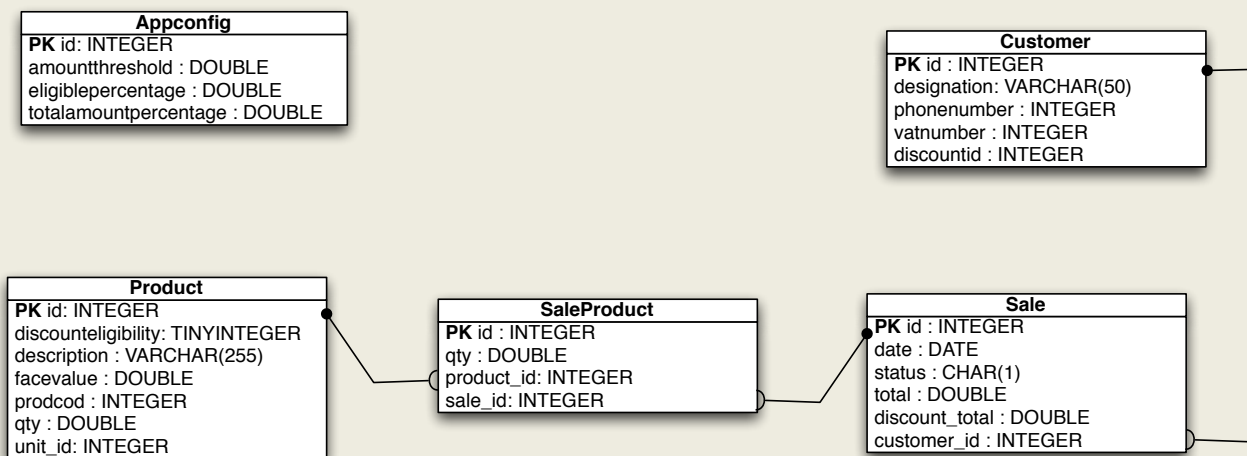
A empresa possui uma carteira de clientes com quem comercializa produtos. Nesta primeira fase, cada cliente é caracterizado por um número de pessoa coletiva, uma denominação (nome do cliente) e um contacto telefónico. Cada produto comercializado pela empresa tem um código que o identifica univocamente, uma descrição e um preço. Um produto tem, em cada momento, uma quantidade de artigos disponíveis em stock.

As vendas registam transações efetuadas com os clientes. Cada venda grava a data em que a operação foi efetuada, o cliente, a lista de produtos (e respetivas quantidades) transacionadas e o valor total. A empresa pratica, nas vendas que efetua, variados tipos de descontos. Assim, cada venda regista também o valor do desconto aplicado.

Atualmente a empresa pratica dois tipos de desconto nas suas vendas: o primeiro resulta da aplicação de uma percentagem sobre o valor global, caso o cliente atinja um determinado valor total na compra; o segundo corresponde a uma percentagem sobre o total dos produtos que estiverem marcados como elegíveis para desconto. O tipo de desconto a que



## SalesSys: Modelo de Dados



## Exercícios (1 e)

---

Suponha agora que se pretende desenhar a camada de negócio do SalesSys recorrendo ao padrão *Transaction Script*. Esta solução deve basear-se numa camada de acesso aos dados desenhada de acordo com o padrão *Row Data Gateway*.

- ii. Elabore um diagrama de interação que defina o comportamento da operação `addCustomer` de acordo com o padrão *Transaction Script*.
- iii. Repita o que fez anteriormente para as operações:
  - `newSale`,
  - `addProductToSale`,
  - `getSaleDiscount`,
  - `closeSale`.

## SalesSys: Solução com *Transaction Script*

---

```
addCustomer(vat,  
            designation,  
            phoneNumber,  
            discountTypeCode)
```



- **Validação**
- **Processamento**

## SalesSys: Solução com *Transaction Script*

---

```
addCustomer(vat,  
  designation,  
  phoneNumber,  
  discountTypeCode)
```

→ :CustomerTransactionScripts

- **Validação**

- Verificar se código do tipo de desconto é válido
- Verificar se o vat é válido
- Verificar se o nome e o telefone estão preenchidos
- Verificar se já há um cliente com o vat dado (*to fail fast*)

## SalesSys: Solução com *Transaction Script*

---

```
addCustomer(vat,  
  designation,  
  phoneNumber,  
  discountTypeCode)
```

→ :CustomerTransactionScripts

- **Validação**

- Verificar se código do tipo de desconto é válido
- Verificar se o vat é válido
- Verificar se o nome e o telefone estão preenchidos
- Verificar se já há um cliente com o vat dado (*to fail fast*)

- **Processamento**

- Assegurar que o novo cliente é guardado de forma persistente

## SalesSys: Solução com *Transaction Script*

```
addCustomer(vat,  
            designation,  
            phoneNumber,  
            discountTypeCode)
```

→ :CustomerTransactionScripts

- **Validação**

- Verificar se código do tipo de desconto é válido
- Verificar se o vat é válido
- Verificar se o nome e o telefone estão preenchidos
- Verificar se já há um cliente com o vat dado (*to fail fast*)

- **Processamento**

- Recorrendo aos serviços prestados pela camada de dados, guardar o novo cliente na base de dados

Customer
<b>PK</b> id : INTEGER
designation: VARCHAR(50)
phonenummer : INTEGER
vatnumber : INTEGER
discountid : INTEGER

## SalesSys: Solução com *Transaction Script*

```
addCustomer(vat,  
            designation,  
            phoneNumber,  
            discountTypeCode)
```

→ :CustomerTransactionScripts

- **Validação**

- Verificar se código do tipo de desconto é válido
- Verificar se o vat é válido
- Verificar se o nome e o telefone estão preenchidos
- **Recorrendo aos serviços prestados pela camada de dados**, verificar se já há um cliente com o vat dado (*to fail fast*)

- **Processamento**

- **Recorrendo aos serviços prestados pela camada de dados**, inserir uma linha na tabela *Customer*

Customer
<b>PK</b> id : INTEGER
designation: VARCHAR(50)
phonenummer : INTEGER
vatnumber : INTEGER
discountid : INTEGER

## SalesSys: Solução com *Transaction Script*

### CustomerTransactionScripts

`addCustomer(int vat, String desg, int phone, int discountCode)`

- Verificar se código do tipo de desconto é válido

`isValidDiscountType(discountCode)`

- Verificar se o vat é válido: `isValidVat(vat)`

- Verificar se o nome e o telefone estão preenchidos:

`isFilled(desig) isFilled(phone)`

- **Recorrendo aos serviços prestados pela camada de dados**, verificar se já há um cliente com o vat dado (*to fail fast*)

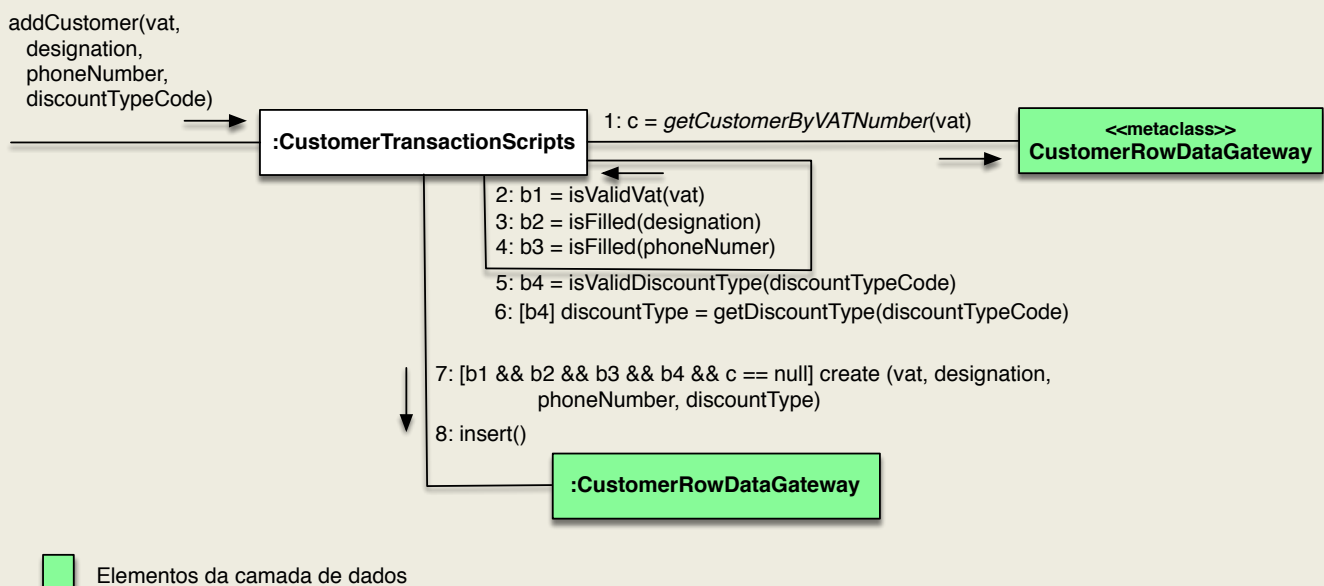
`CustomerRDGW.getCustomerByVatNumber(vat) != null`

- **Recorrendo aos serviços prestados pela camada de dados**, inserir uma linha na tabela *Customer*

`new CustomerRDGW(vat, desg, phone, getDiscountType(discountCode)).insert()`

Customer	
PK id	: INTEGER
designation	: VARCHAR(50)
phonenumber	: INTEGER
vatnumber	: INTEGER
discountid	: INTEGER

## SalesSys: Diagramas de Interação





## SalesSys: Implementação

```
public void addCustomer(int vat, String denomination, int phoneNumber, int discountCode)
    throws ApplicationException {
    // Checks if it is a valid VAT number
    if (!isValidVAT(vat))
        throw new ApplicationException("Invalid VAT number: " + vat);

    // Checks if the discount code is valid.
    if (discountCode <= 0 || discountCode > DiscountType.values().length)
        throw new ApplicationException ("Invalid Discount Code: " + discountCode);

    // Checks that denomination and phoneNumber are filled in
    if (!isFilled(denomination) || phoneNumber == 0)
        throw new ApplicationException(
            "Both denomination and phoneNumber must be filled");

    /*
    Tries to store the customer in the database. There is a
    unique key constraint associated with the VAT number, so if the
    VAT number already exists in the database a persistence exception will
    be thrown.*/
    try {
        CustomerRowDataGateway newCustomer = new CustomerRowDataGateway(vat, denomination,
            phoneNumber, DiscountType.values()[discountCode-1]);
        newCustomer.insert();
    } catch (PersistenceException e) {
        throw new ApplicationException("Error inserting the customer into the database", e);
    }
}
```



ULisboa

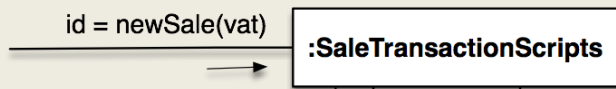
## SalesSys: Solução com *Transaction Script*



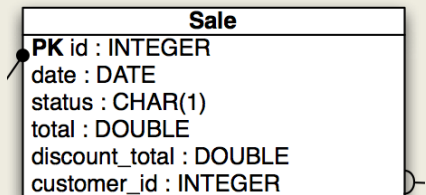
- Validação
- Processamento



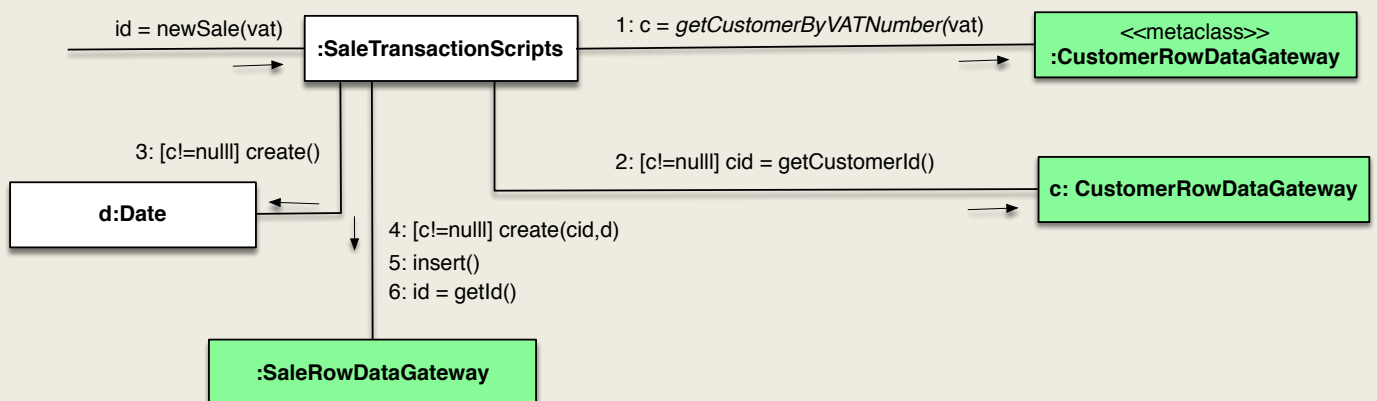
## SalesSys: Solução com *Transaction Script*




- Validação
  - **Recorrendo aos serviços prestados pela camada de dados**, verificar se há um cliente com aquele vat
- Processamento
  - **Recorrendo aos serviços prestados pela camada de dados**, registar uma nova venda com a data corrente na base de dados; notar que isto implica ter de obter a chave do registo do cliente com o vat dado
  - Devolver o número de registo da venda

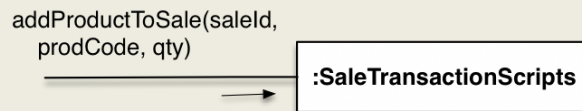


## SalesSys: Diagramas de Interação



 Elementos da camada de dados

## SalesSys: Solução com *Transaction Script*



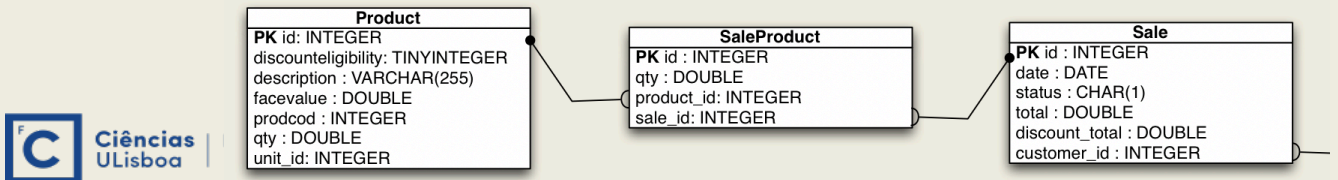
- **Validação**

- **Recorrendo aos serviços prestados pela camada de dados**, verificar se

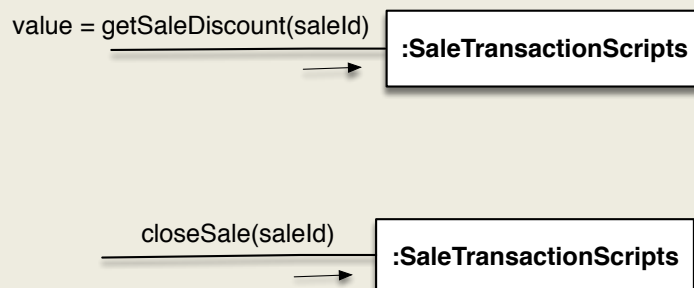
- existe venda com identificador saleId e não está fechada
    - existe produto com código prodCode
    - verificação do stock do produto

- **Processamento**

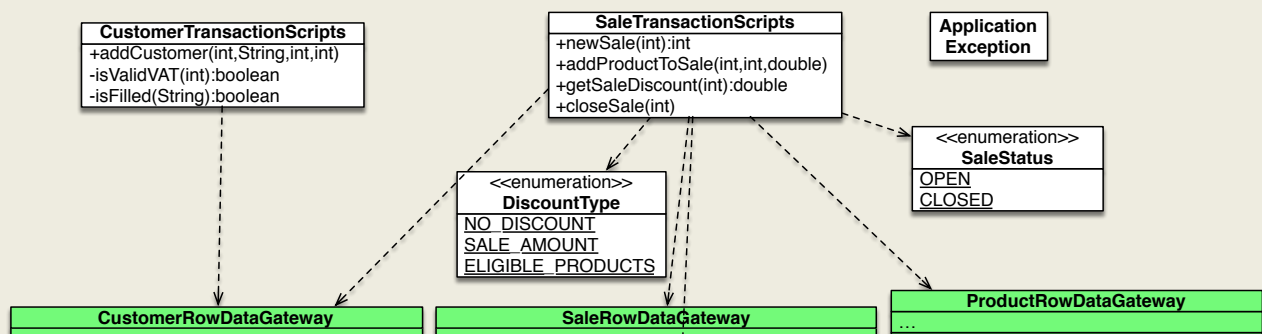
- registar nova entrada na venda; notar que isto implica ter de obter a chave do registo do produto com o código dado
  - Registrar alterações no stock do produto




## SalesSys: Diagramas de Interação



## SalesSys: Modelo de Classes



 Elementos da camada de dados