TÉCNICO LISBOA

# Integração Empresarial

2º semestre - 2016/2017

## 2ª Entrega -
## Service Trees e BPMN

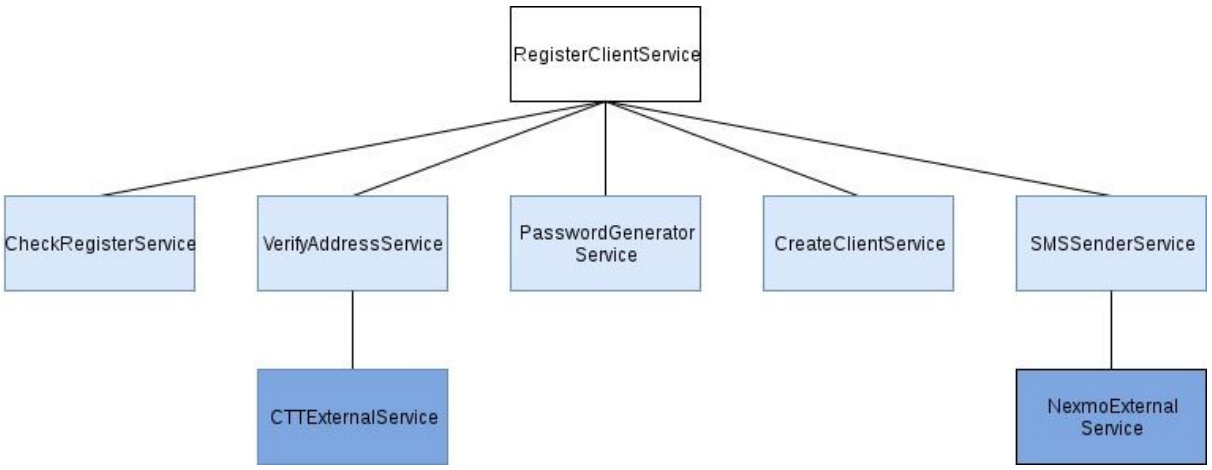

**Grupo 9**

79023 - Carolina Fernandes

79055 - Pedro Ribeiro

Este relatório apresenta a arquitetura orientada a serviços (SOA) e Processos de Negócio (BPMN) seguidos para o desenvolvimento da *FormoPlace* (*Formo* é o latim para *moda*).

**1. Register Client Service Tree**

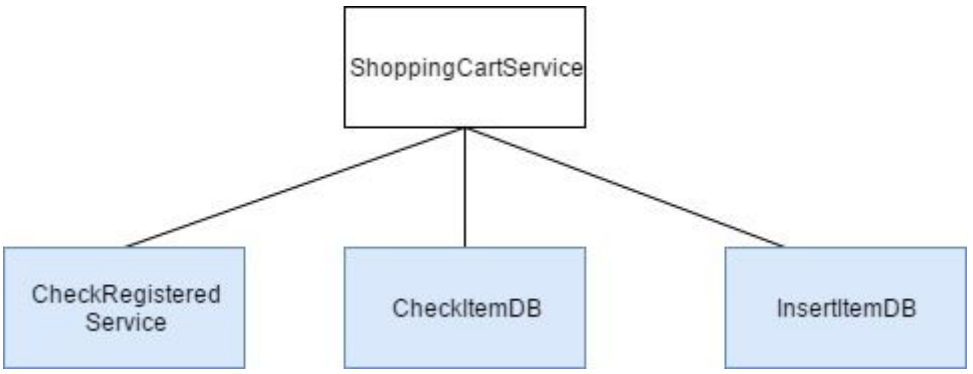



Para a implementação deste serviço, foram usados os seguintes serviços:

- **RegisterClientService:**
  - (String Email, String Address, String PhoneNumber) -> (String Response)
  - Tenta criar um utilizador, caso este não exista, e introduzi-lo na base de dados *client*. Para tal, atribui-lhe um user id e gera uma password aleatória, a qual é enviada ao cliente através de um SMS. Retorna se teve ou não sucesso.
- **CheckRegisterService:**
  - (String Email) -> (String Response);
  - Usa um adaptador para aceder à base de dados de modo a validar se um cliente está registado na base de dados *client*.
- **VerifyAddressService**:
  - (String Address) -> (Boolean IsOk);
  - Utiliza o serviço externo dos CTT para validar se a morada introduzida é válida.
- **PasswordGeneratorService**:
  - ( ) -> (String Password);
  - Gera uma password alfanumérica aleatória com um tamanho específico (8 caracteres).
- **CreateClientService**:
  - (String Email, String Password, String Address, String PhoneNumber) -> ( );
  - Usa um adaptador para aceder à base de dados de modo a criar um novo cliente na base de dados *client*.
- **SMSSenderService**:
  - (String To, String PhoneNumber) -> (boolean isOk);

- Utiliza o serviço externo do Nexmo para mandar uma SMS de confirmação para o número indicado no registo, com o id do novo cliente e a password gerada.

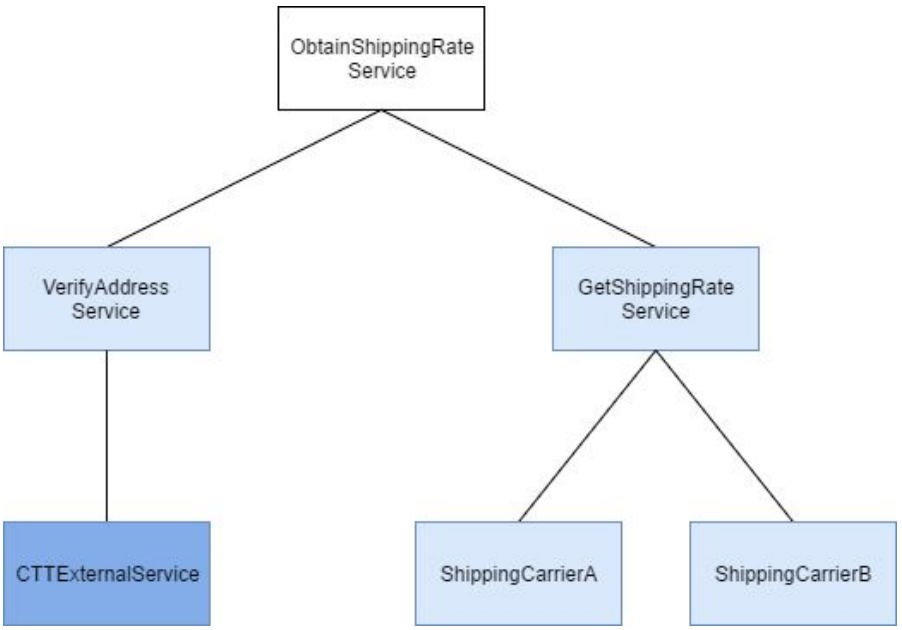
**2. Shopping Cart Service**




Para a implementação deste serviço, foram usados os seguintes serviços:

- **ShoppingCartService:**
  - (int Userid, String password, int Itemid, int Quantity) -> (String Response);
  - Tenta inserir um item no carrinho de compras de um utilizador registado. Retorna se teve sucesso ou não.
- **CheckRegisteredService:**
  - (int Userid, String password) -> (boolean isRegistered);
  - Usa um adaptador para aceder à base de dados de modo a validar se um cliente está registado na base de dados *client*.
- **CheckItemDB**:
  - (int Itemid) -> (String Response);
  - Adaptador para verificar se um determinado produto existe na base de dados *stock*.
- **InsertItemDB:**
  - (int Userid, String password, int Itemid, int Quantity) -> ( );
  - Adaptador para aceder à base de dados *cart* e inserir o produto no carrinho de compras.

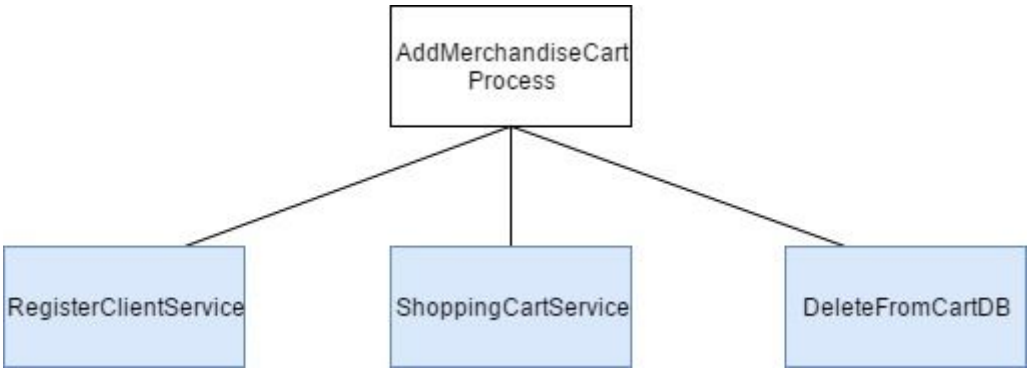
### 3. Request shipping rate Service Tree




Para a implementação deste serviço, foram usados os seguintes serviços:

- **ObtainShippingRate:**
  - (String boutiqueAddress, String clientAddress, int shippingTimes) -> (String carrierName, double shippingRate, string status);
  - Verifica se a morada de envio é válida e, caso seja, calcula entre dois providers A e B qual o que oferece um preço mais baixo. Retorna se teve sucesso ou não.
- **VerifyAddressService**:
  - (String Address) -> (Boolean IsOk);
  - Utiliza o serviço externo dos CTT para validar se a morada introduzida é válida.
- **GetShippingRateService**:
  - (String boutiqueAddress, String clientAddress, int shippingTimes) -> ( double shippingName, String carrierName);
  - Compara as ofertas das duas shipping carriers (A e B) e devolve a melhor taxa e o respectivo Shipping Carrier. Invoca os serviços de ShippingCarrierA e ShippingCarrierB, que calculam um número random entre 2 algarismos correspondente à taxa.

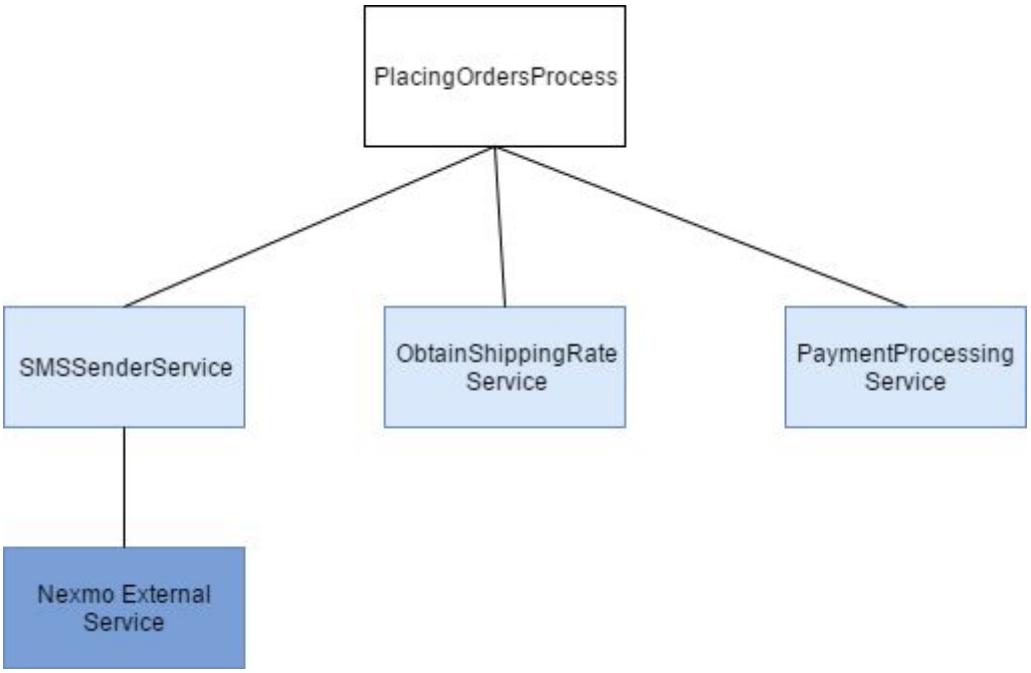
## 4. Adding Merchandise Process Tree




Este processo reutiliza serviços já implementados (e descritos nas primeiras páginas deste relatório) de modo a permitir utilizadores registados introduzir merchandise no carrinho de compras.

- **DeleteFromCartDB:**
    - (int Userid, int Itemid) -> ( );
    - Adaptador que elimina da base de dados *cart* a entrada relativa ao produto Itemid.

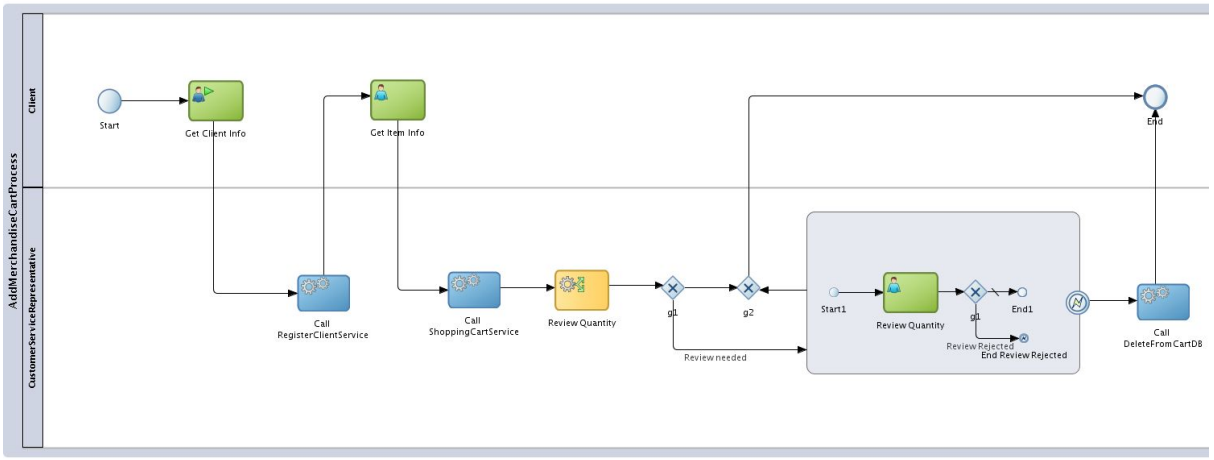
## 5. Placing Orders Process Tree




Para evitar a repetição, não foi adicionada à árvore a continuação do ObtainShippingRate Service que está representada no ponto 3 do relatório..

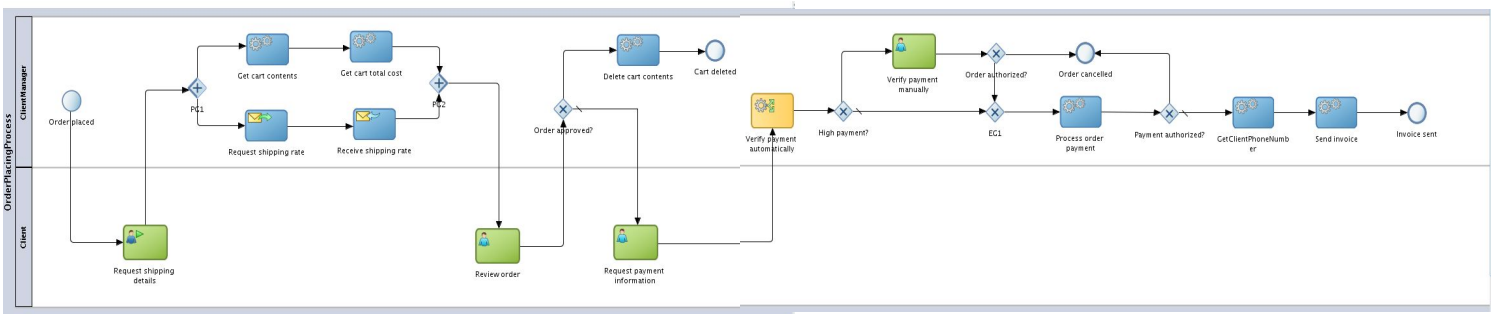
- **SMSSenderService**:
    - (String To, String PhoneNumber) -> (boolean isOk);
    - Utiliza o serviço externo do Nexmo para mandar uma SMS de confirmação para o número indicado no registo, com o id do novo cliente e a password gerada.
- **ObtainShippingRate:**
    - (String boutiqueAddress, String clientAddress, int shippingTimes) -> (String carrierName, double shippingRate, string status);
    - Verifica se a morada de envio é válida e, caso seja, calcula entre dois providers A e B qual o que oferece um preço mais baixo. Retorna se teve sucesso ou não.
- **PaymentProcessing:**
    - ( int credit card, double value) -> (boolean)
    - Simula um sistema de pagamento. Tem 25% de falhar um pagamento.

Processo **Add Merchandise Cart**:




1. São pedidos os dados que serão usados na invocação ao serviço RegisterClient.
2. São pedidos dados que serão usados na invocação ao serviço ShoppingCart. No final o item irá ser introduzido no carrinho de compras.
3. A regra de negócio é validada. Se a quantidade for menor que 50 o processo termina.
4. Caso contrário, o *Customer Service Representative* irá analisar a tentativa e escolher se pode ser aceite ou não.
5. Se for aceite, o processo termina com o item adicionado.
6. Se não for, é lançado um erro e o item eliminado.

Processo **Placing Orders**:




1. É pedida a informação de shipping ao cliente;
2. É obtido o conteúdo e custo total do carrinho e, em paralelo, é invocado o serviço Obtainshippingrate;
3. O cliente revê os itens adicionados ao seu carrinho, se aprovar a encomenda, são-lhes solicitadas as informações de pagamento, como o NIF, número de cartão de crédito e morada de faturação. Se não aprovar, é apagado todo conteúdo do carrinho e o carrinho é eliminado;
4. Após terem sido enviadas as informações de pagamento, é verificado o pagamento automaticamente através de uma business rule que deteta pagamentos muito elevados, ou seja, maior ou igual a 100, que são verificados manualmente.
5. Se a encomenda não for autorizada é cancelada, se for autorizada é feito o processamento do pagamento da encomenda;
6. Se o pagamento da encomenda não for autorizado, é cancelada a encomenda. Se for autorizado, através de um database adapter vai-se buscar o número de telefone do cliente à Base de dados;
7. Por fim, é enviada a fatura ao cliente através do serviço SMSSender , e o processo termina com o envio da fatura ao cliente, por sms.