A aplicação tem como funções a implementação de um serviço de cobrança e também a função de consulta por MSISDN.

Serviço de Cobrança:

Como a aplicação tinha como ponto de partida a receção de um ***ChargingRequest***, e o mesmo não estava definido como iria ser feito, parti do princípio da inserção de valores pelo utilizador via consola.

Sendo assim, o processo deste sistema, começa com o pedido de inserção de diferentes valores e pela posterior validação dos mesmos. Em caso de valores incorretos, o processo continua até serem colocados todos os valores de acordo com o pretendido.

Estando os valores validados, o próximo passo é a criação de um ***RequestedServiceUnit***, com os valores previamente inseridos. Esta classe estende a *superclass* ***ServiceUnits*** já que podemos usar aqui o princípio da herança. Este princípio foi utilizado, porque a subclasse partilha os mesmos princípios que a classe pai, podendo tonar-se mais especifica se for necessário. Existe uma outra classe chamada ***GrantedServiceUnits*** que também é subclasse da classe principal, sendo utilizada a mesma lógica para esta subclasse.

O segundo passo foi então a criação do ***ChargingRequest***. Esta classe contém os atributos pedidos no modelo. Um destes é o objeto ***RequestedServiceUnit***, que é passado via construtor e posteriormente armazenado numa variável encapsulada. Assim sendo, temos acesso aos valores requisitados via getter quando utilizado o objeto instanciado.

A seguir foi a criação do objeto ***BillingAccount***. Este objeto foi criado antes do serviço, visto que é necessário ter valores de *bucket* (valores random), para posteriormente serem usados na procura do melhor tarifário. Para criar este objeto, foi necessário passar o valor de MSISDN, via construtor, através de um get. Este valor vem do objeto ***ChargingRequest*.** No objeto atual, as variáveis estão encapsuladas, para não permitir o acesso direto, proteger o código e também para permitir fazer o set das variáveis de *counter* posteriormente aquando do fim da criação do tarifário.

Findando este processo, fazemos então a criação de um serviço. Para a criação do serviço foi necessário a passagem de alguns valores e do objeto ***BillingAccount*** para posteriormente serem utlizados para descobrir o melhor serviço e tarifário. Existem neste objeto valores *static* que ao serem posteriormente incrementados consoante a necessidade, vão permitir atualizar os valores no objeto ***BillingAccount***. Para descobrir qual o melhor tarifário para as necessidades do utilizador é utilizado o método ***findTariffByService***. Aqui é instanciado um objeto do tipo *Alpha* ou *Beta*, consoante o tipo de serviço passado via construtor. Ambos estes objetos estendem uma classe abstrata chamada ***Tariff***, visto que ambos os objetos partilham a mesma base. Desta forma podemos reutilizar código, utilizar polimorfismo, podendo partir de uma base comum para posteriormente especificar em cada um dos objetos estendidos.

Como temos o valor do serviço (A/B) podemos criar um objeto *Tariff* do tipo *Alpha/Beta*. Em ambos os casos o processo de procurar qual o melhor tarifário dentro do serviço pretendido é semelhante. Existem vários métodos com funções especificas. O primeiro passo é verificar quais destes são elegíveis, depois é guardado num *hash map* os serviços e o seu respetivo valor unitário. O valor cobrado tem em conta as especificidades de cada tarifário. Daqui em diante é verificado qual o serviço com menor valor e é retornado o valor para verificar se é possível retirar do respetivo *bucket*. Em caso de sucesso é

retornado um objeto com os dados do respetivo serviço, podendo o resultado variar entre ***OK*** e ***CREDIT_LIMIT_REACHED***. Este último acontece quando o valor do respetivo *bucket* não é superior ao valor total do tarifário. Em caso de não haver um tarifário que corresponda aos critérios é retornado um valor ***null***, com o resultado ***NOT_ELIGIBLE***.

Retornando um objeto ***Service***, é verificado caso tenha um tarifário ***null*** ou não. Caso não seja nulo, é então criado um objeto ***GrantedServiceUnits*** com os valores do ***ChargingRequest***, mais os valores do resultado do serviço. Como dito anteriormente, este objeto estende a superclasse ***ServiceUnits***.

Seguidamente é criado um ***ChargingReply***, com o identificador do pedido, o resultado do serviço e com o objeto ***GrantedServiceUnits***.

Tudo isto para no fim criar um ***ClientDataRecord*** com a informação pedida no modelo de dados. Este objeto serve para guardar a informação e mostrar o resultado da aplicação.

Consulta e Relatórios:

Para facilitar a vista deste tópico, já vem incluído no código um conjunto de CDR para diferentes MSISDN e serviços de forma a mostrar a o uso e os custos associados a cada um destes, ordenado pelo *timestamp*.