Welson de Lima Teles – 47932

O projeto consiste em um protótipo bem básico de um autorizador bancário que contempla algumas funcionalidades: saque, deposito, transferência entre contas, extrato e saldo.

São três as principais classes, nas quais, são abstratas: *AbstractAutorizador*, *AbstractTransacao* e *AbstractAutorizacao*. Para adicionar uma nova funcionalidade, é preciso criar classes estendendo essas classes, e colocado a regra negocial específica de cada funcionalidade em cada uma delas no metodo *executaRegrasEspecificas*. As classes que forem estender de *AbstractAutorizador* e *AbstractAutorizacao d*evem se anotadas com *@Transacao* informando a funcionalidade a qual ela pertence.

Sobre o funcionamento, basicamente precisa criar uma transação, o autorizador processará e retornar uma autorização dizendo se a transação foi realizado com sucesso, ou se falhou, e nesse caso ele informará o por que da falha.

O sistema de injeção de dependências se responsabilizará pela instanciação dessas classes anotadas com *@Transacao*, portanto é necessário informar estas na classe *RegistroClassesDI*, para que isto ocorra corretamente.

AbstractAutorizador

A classe *AbstractAutorizador* tem os seguintes métodos: *AbstractAutorizacao executa*(*AbstractTransacao*), *abstract void executaRegrasEspecificas*(*AbstractTransacao*, *AbstractAutorizacao*), *Conta getContaCliente*(*AbstractTransacao*) e *getContaService*().

O método executa é o principal, ele executa todo o código comum a todos autorizadores. Nele também é chamado o método *executaRegrasEspecificas*, este, deve ser implementado nas classes filhas, e que dá um comportamento personalizado a classe *AbstractAutorizador*.

AbstractAutorizacao

Contem todas os atributos comuns de uma resposta de uma transação, e as classes filhas têm os atributos específicos, por exemplo, em uma transação de extrato, é preciso responder uma lista de lançamentos.

AbstractTransacao

Contem todos os atributos comuns que precisam ser enviados em uma transação.

DAOs

São as classes que fazem a persistência dos objetos, no caso desse projeto, ele serializa os objetos em arquivos.

UML