Utilização de Padrões

Conteúdo

Introdução

Padrões de projeto ou Design Patterns são técnicas utilizadas para tornar o código, de projetos orientados a objeto, mais reutilizavel melhorando sua manutenabilidade. Os padrões de projeto são modelos de soluções que podem ser utilizados em diversos contextos. Seguindo a refência do GoF, os padrões de projeto podem ser: estruturais (tratam das associações entre classes e objetos), criacionais (relacionados à criação de objetos) e compartamentais (tratam das interações e divisões de responsabilidades entre as classes ou objetos). Nesse projeto foram utilizados vários padroes de projeto de diferentes naturezas. No decorrer do documento segue uma breve explicação do padrão uma justificativa pelo uso ou não uso do padrão e se for o caso de utilização um trecho de código que mostra essa utilização.

Em uma outra perspectiva padrões arquiteturais seguem a linha de organização ligada a estrutura do sistema. Os pardões arquiteturais definem como o sistema vai se comportar em relação a abordagem escolhida pelo arquiteto, cabe a ele ver qual das abordagens disponiveis se adapta melhor para determinada situação. Dentro desse projeto se utilizou alguns padrões arquiteturais como por exemplo o MVC.

Padrões de Projeto

Singleton

O padrão singleton é um padrão estrutural que é utilizado para que uma classe possa ter apenas uma instancia. Para isso o construtor da classe é selecionado como privado e é disponibilizado um método que vai conter a instancia dessa classe para acessa-la.

No nosso projeto o singleton foi utilizado na classe de Faces do framework JSF para que fosse criada apenas uma instancia da classe FaceContext, o código abaixo mostra uma chamada dessa classe

```
FacesContext.getCurrentInstance().addMessage("form", msg);
```

Esse código está avaliado no arquivo LoguinMB, linha 26.

Adapter

O padrão adapter, também chamado de Wrapper, é utilizado para que diferentes classes que possuem comportamento semelhantes sejam utilizadas da mesma forma, servindo como um 'adaptador'.
No nosso projeto, o padrão adapter foi utilizado na execução da funcionalidade de login, fazendo com que a classe LoginMB possuisse um método 'logar()' que chama o método 'logar()' da classe LoginBean, passando por parâmetro os próprios atributos do objeto LoginMB.

```
public String logar(){
        Usuario usuarioLogado = loginBean.logar(login, senha);
            System.out.println("Usuario logado: "+usuarioLogado);
            if (usuarioLogado != null){
                    return "OK";
            }
            FacesMessage msg = new FacesMessage("Usuário ou senha
incorretos");
     FacesContext.getCurrentInstance().addMessage("form", msg);
     return "";
    }
```

Esse padrão pode ser encontrado no arquivo LoginMB.java, na linha 13.

State

O padrão State é um padrão comportamental, que serve para variar comportamentos que estão diretamente ligados ao estado de um objeto. Permite também a inserção de novos comportamentos, sem que seja necessário editar o código existente.
Seria utilizado nos estados das despesas: PAGO e PENDENTE.

Observer

O padrão Observer é um padrão comportamental que funciona como um notificador que aciona um evento, que por sua vez é 'escutado' por um ou mais 'listener', promovendo a execução de tarefas baseadas em eventos.
No nosso projeto, esse padrão foi utilizado no funcionamento interno do JSF, que promove a execução de operações baseadas nos elementos de interface gráfica. Como no exemplo abaixo:

```
<h:commandButton id="loginButton" value="Entrar" action = "#{loginMB.logar}"/>
```

Esse padrão pode ser observado no arquivo login.xhtml , linha 25.

Factory

O padrão factory cria objetos sem que se saiba a classe específica a ser instanciada. Delega a criação dos objetos para subclasses. No projeto em questão não foi utilizado o padrão factory visto que com a utilização do EJB o framework manipulou a parte de ciração de objetos.

Facade

O padrão Facade é um padrão estrutural, que funciona como uma faixada que se responsabiliza pela comunicação de diversas classes, executando determinadas operações, tornando invisível para a classe cliente o que realmente está sendo executado.
Decidimos não utilizar esse padrão em nosso projeto devido a pouca complexidade existente nas operações do sistema. A utilização do mesmo para sistemas que possuem

pouca complexidade de operações, torna-se inviável.

◦ Iterator

O padrão Iterator ou iterador é um padrão que fornece uma ferramenta ao programador para iterar em cima de uma coleção. Esse padrão, por conta da sua forma de utilização, já é implementado no Java de maneira implicita, dispensando o serviço do programador de criar o iterator.
No nosso projeto o padrão iterator foi utilizado para manipular o retorno do banco de dados que vem da consulta dos das receitas de determinados usuários:

```
            for (Receita receita: receitasMes) {
  System.out.println(receita);
 }
```

Esse trecho de código pode ser avaliado em ReceitaBS.java, linha 36.

Padrões de Arquitetura

◦ MVC

O padrão Model Vlew Controller (MVC) é um padrão arquitetural que separa trata da forma de iteração do usuário com o sistema, dividindo o acesso a informações ,lógica de négocio e apresentação ao usuário em partes diferentes. Nesse projeto se está utilizando para realizar a parte de comunicação com o usuário o framework JSF. A utilização dessa plataforma de desenvolvimento proporciona a utilização desse padrão por possuir um suporte para a manipulação de eventos e organização dos componentes permitindo que a parte a ser implementada deve ser apenas a da logica de negocio.

◦ Camadas

O desenvolvimento em camadas consiste em dividir o projeto em algumas camadas, geralmente três, e atribuir alguma responsabilidade a cada uma das camadas. Por exemplo se em um sistema existem classes responsaveis pela persistencia e outras ligada ao negocio então a divisão que existe é em camadas com a camada de persistência e a camada de negocios. Nesse projeto adotou-se a divisão em camadas com as classes referentes a camada de apresentação no webcontent (páginas xhtml), negocio as classes associadas ao pacote business e as relacionadas a persistencia estão integradas no pacote dao.