Descrição dos Padrões de Projeto implementados no projeto DouShouQi

Kevin Kons[1]

[1]Departamento de Engenharia de Software
Universidade Estadual de Santa Catarina (UDESC) – Ibirama, SC – Brazil

kevin.kons@hotmail.com

Abstract. *This meta-paper describes the utilization of the Design Patterns implemented in the first delivery of the game DouShouQi implemented to the Design Patterns subject. The Design Patterns applied were: MVC, Singleton, Observer, AbstractFactory, Command and Builder.*

Resumo. *Este meta-artigo descreve a utilização dos Padrões de Projeto implementados na primeira entrega do jogo DouShouQi implementado para a disciplina de Padrões de Projeto. Os padrões aplicados foram: MVC, Observer,Singleton, AbstractFactory, Command e Builder.*

1. MVC (Model-View-Controller)

Para a divisão do projeto em camadas bem definidas foi utilizado o padrão arquitetural MVC. As classes que estão no pacote ”model”fazem parte da camada Model, com a utilização dessa camada é possível isolar as regras de negócio. As classes do pacote ”controller”constituem a camada Controller, a qual define o fluxo das apresentações funcionando como uma conexão entre regras de negócio e camada de apresentação que por sua vez encontra-se no pacotem ”view”e separa toda a logíca de apresentação do resto da aplicação.

2. Observer

Para tornar possível a comunicação entre a camada de controle e a camada de apresentação foi utilizado o padrão Observer, por meio dele a camada de controle notifica a camada de apresentação sempre que alguma informação é alterada. Na implementação desse padrão temos a necessidade de duas interfaces, a Observado que é implementada pela classe Controller e a Observador que é implementada pela classe TelaPrincipal. A classe TelaPrincipal na linha 52 se define como uma observadora da classe Controller. Nas linhas 28, 37, 48, 55, 70 e 73 da classe TelaPrincipalController é realizada a chamada dos métodos do Observador. Já nas linhas 102 e 109 estão respectivamente os métodos para inclusão e exclusão de Observadores.

3. Singleton

A aplicação possui uma classe chamada Tabuleiro esta trata-se de um Singleton pois necessita-se apenas uma instância dessa classe ao longo da execução da aplicação. O ponto de acesso global da classe Tabuleiro encontra-se na linha 21 e é chamado na linha 25 da classe ”TelaPrincipalController”, a variável estática que guarda sua instância na linha 19, e na linha 29 um método chamado ”newInstance()”para definir uma nova instância, reiniciando então o tabuleiro, a chamada para esse método está na linha 34 da classe ”TelaPrincipalController”.

4. Abstract Factory

Este padrão foi aplicado em duas partes da aplicação, uma para a criação de terrenos e outra para a criação de tocas. Na primeira situação temos a classe “FabricaAbstrataDeTerreno” que define a interface para criação dos terrenos e a classe “FabricaDeTerreno” que herda da classe anterior, realiza a criação dos terrenos e está sendo instanciada na linha 46 da classe ”Tabuleiro”, no método “montarTabuleiro” dessa mesma classe e nos métodos “constroiArmadilha” e “constroiToca” da classe “BuilderTimeA” e “BuilderTimeB” estão localizadas a chamada dos métodos de criação de terrenos. Já na segunda situação temos a classe “FabricaAbstrataDePeca” que define a interface para a criação das peças e a classe “FabricaDePeca” que herda da classe “FabricaAbstrataDePeca”, realiza a criação das peças e é instanciada na linha 45 da classe “Tabuleiro”. A chamada dos métodos de criação de peças se da nos métodos de construção de peças das classes “BuilderTimeA” e “BuilderTimeB”.

5. Command

Para realizar a seleção e a movimentação de peças foi utilizado o padrão Command que encapsula a solicitação de seleção na classe “SelecionarPeca” e a solicitação de movimento na classe “Movimentar”, ambas as classes implementam a interface “Command”

que define um método de execução. A chamada desse método ocorre na classe “CommandInvoker” que é instanciada na classe “TelaPrincipalController” na linha 21 e decide qual solicitação executar, as solicitações de execução encontram-se na mesma classe nas linhas 52 e 67.

6. Builder

Na criação dos objetos Time foi utilizado o padrão Builder que separa a criação desse objeto complexo na classe “BuilderTime”, para a criação das peças e dos terrenos que fazem parte de um time são utilizados objetos que herdam de “FabricaAbstrataDePeca” e de “FabricaAbstrataDeTerreno”, ambos são recebidos como parâmetros na instanciação de um descendente de “BuilderTime” o qual pode ser um “BuilderTimeA” que é instanciado na linha 56 da classe “Tabuleiro” ou um “BuilderTimeB” que é instanciado na linha 63 da classe “Tabuleiro”. A classe que coordena a ordem de execução dos métodos de construção se chama ”Tecnico”e é instanciada nas linhas 57 e 64 da classe “Tabuleiro” e seu método de construção é chamado nas linhas 58 e 65 dessa mesma classe.