INSTITUTO FEDERAL DO ESPIRITO SANTO
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

RELATÓRIO DE PROJETO
XADREZ

DIEGO PASTI
Serra, 8 de junho de 2014

Relatório de Projeto

Esse documento demonstra os principais conceitos e tecnologias utilizadas no projeto Xadrez, ministrada na disciplina de Programação Orientada a Objetos II. Abaixo segue uma breve explicação de como o projeto foi estruturado.
Repositório do Projeto: https://github.com/diegopasti/faculdade-poo2-xadrez
Site do Projeto: http://diegopasti.github.io/faculdade-poo2-xadrez/

Padrão Arquitetural Modelo-Visão-Controle

Na fase de projeto, o sistema foi dividido seguindo dois estilos arquiteturais sendo eles o estilo Partição e Camadas. Por opção e circunstancias (de tempo e até da estrutura herdada) foi decidido manter apenas uma partição e organizar a estrutura interna em camadas.
Para estrutura das camadas foi utilizado o padrão Modelo-Visão-Controle e suas subdivisões, sendo elas: Camada de Interação Humana, Camada de Controle de Interface, Camada de Domínio do Problema, Camada de Gestão de Tarefas e Camada de Gestão de Dados. Essa estrutura permite que o projeto seja organizado de acordo com os tipos de tarefas, que basicamente são:

Camada	Subcamada	Descrição
Modelo	Gestão de Tarefas	Camada responsável pelo comportamento das entidades do projeto (ex. casos de usos do sistema).
	Domínio do Problema	Camada responsável por representar todas as entidades envolvidas no projeto.
	Gestão de Dados	Camada responsável por gerenciar a manipulação da base de dados.
Visão	Interação Humana	Camada responsável por reunir as classes que manipulam a interface.
Controle	Controle de Interação	Camada Responsável por ser uma ponte entre a comunicação dos modelos e as visões.

Conceitos utilizados

Todo o projeto foi desenvolvido utilizando a linguagem Java e essencialmente a orientação à objetos, seguindo os princípios de programação voltada à interface, priorizando o uso de composições a heranças. Nenhuma variável global foi utilizada, os padrões de nomenclatura da linguagem Java foram mantidos e boas práticas de codificação seguindo o padrão clean code também foram seguidos. O resultado foi códigos mais fáceis de cuidar e manter agregando valor ao produto final. Recursos específicos da linguagem tais como Generics (na utilização de listas) e Reflection(na utilização de acesso a arquivos de imagem do projeto) também foram utilizados.

Projeto

Primeira Etapa

A versão atual do sistema não apresenta nenhum problema relacionado a compilação. Foi detectado um erro em tempo de execução em um caso muito especifico e relativamente raro, mas que a solução (até simples por sinal) ainda não pode ser implementada. A lógica do projeto está bem coesa, contudo pontos específicos como tipo de movimento da peça podem ser aprimoradas (e serão) e funcionalidades como desistir, pedir empate e o cadastro dos jogadores estão incompletas. Os diagramas do sistema estão aproximadamente 70% a 80% atualizado, um numero significativo de modificações foram feitas nos últimos dias e tais modificações ainda não foram reproduzidas nos modelos.

Todos os tipos de movimentos de peças xeque e xeque-mate foram implementados, jogadas específicas como El Passant, promoção, roque menor e maior não foram implementadas.

Devido um conjunto de fatores a ferramenta Maven não pode ser configurada e utilizada no projeto, com isso, toda e qualquer solução ou técnica que a exigisse também não pode ser implementada, dentre elas o armazenamento via Hypersonic dos dados e a análise de qualidade de código via Sonar.

O projeto contabiliza a pontuação dos jogadores em cada partida (apesar do placar ter sido suprimido nessa versão – pelo menos em sua forma como era em versões passadas) o que não está implementado ainda é o registro de vitórias, derrotas e empates para os jogadores nem o ranking de pontuação dos mesmos.

O projeto pode ser jogado por 1 ou 2 pessoas, a lógica para as jogadas do computador(Zeus) são simples mas funciona satisfatoriamente (possíveis modificações estão previstas para sua melhoria) contudo não é possível interromper o jogo e continuar posteriormente, a lógica para implementação desta tarefa não foi realizada já o Chat para conversação dos jogadores foi realizada mas não há como (ainda) persistir as mensagens trocadas.

Não foram utilizados testes unitários para controle de qualidade. Os padrões de projeto utilizados são descritos na próxima sessão.

Segunda Etapa

Para a segunda etapa do projeto que seria basicamente a inclusão dos padrões comportamentais foi corrigido e complementado alguns dos requisitos de implementação que faltaram na primeira etapa.

Padrões de Projeto

O projeto foi estruturado seguindo alguns padrões de projeto criativo e adicionalmente alguns padrões comportamentais. Abaixo segue uma breve explicação da utilização de cada um no projeto:

- **Fabrica Abstrata**: Foi utilizado para a instanciação do controle de peças (brancas e pretas) permitindo que a instanciação de cada peça feita seja chamada em apenas um ponto dentro do código, e devido a interface utilizada e os conceitos de polimorfismo

podemos alterar uma fabrica de peças pretas ou brancas sem termos que alterar os métodos como são invocados.

- **Protótipo**: Foi utilizado nos slots (casa) do tabuleiro, apesar da sua utilização ter sido parcialmente suprimida devido a não implementação manual da clonagem de atributos complexo como listas. Com ela iríamos instanciar apenas uma vez o slot, e o utilizaríamos como modelo para clonagem dos demais slots que é muito mais eficiente.
- **Observador**: O padrão comportamental observador foi utilizado para notificarmos alterações na matriz de peças do tabuleiro. Para tal cada peça implementa uma interface observável e recebe ao ser instanciada o objeto que irá monitorar e notificar as mudanças do tabuleiro.

- **Estratégia**: O padrão comportamental estratégia foi utilizado para definirmos a lógica de movimentação das peças, visto que este era a grande e praticamente única diferença funcional das peças, dessa forma a metodologia de movimentação de cada peça é um atributo (que pode inclusive, ser alterada em tempo de execução) e implementa uma interface Tipo de Movimento que permite que todos os movimentos sejam invocados da mesma forma. Também foi utilizado para definir a inteligência e a dificuldade do qual o jogador Zeus (maquina) executa, sendo a inteligência as estratégias de jogo (jogadas ofensiva, defensivas e padrão) que ele deve considerar em cada escolha e a dificuldade será analise de jogada com estudo de possíveis jogadas a frente.

Conclusão

Abaixo resumo principais aspectos descritos acima em uma tabela. As cores verde e vermelhas indicam se determinado requisito foi atendido ou não (sendo verde totalmente atendido ou vermelho para incompleto ou não).

Requisitos do Projeto (principais)				
Requisitos Funcionais	Movimento de Peças	Permitir Jogadas Especiais	Pontuação na Partida	Ranking de Vitórias
	Permitir pausar e continuar jogo	Pedir derrota ou empate	Jogar contra a Maquina	Jogar contra outro Jogador
Requisitos de Implementação	Utilizar generics	Utilizar reflection	Utilizar MVC	Utilizar Maven
	Utilizar padrões Criativos	Utilizar padrões Estruturais	Armazenamento com Hypersonic	Clean code
	Orientação a objeto	Utilizar Testes Unitários	Utilizar Sonar	
Requisitos não Funcionais	Utilizar Swing	Utilizar Interface amigável	Utilizar Chat para conversas	Utilizar Banco de Dados

A utilização dos conceitos de orientação a objetos, desenvolvimento voltado à interface, utilização de composição ao invés de heranças, os padrões de projeto e arquitetural, controle de versão, qualidade de código e modelagem permitiram que o projeto pudesse ser desenvolvido com um foco em melhorar organização e manutenabilidade do código alem do próprio desempenho. A utilização destes conceitos impacta muito no que diz respeito à complexidade da tarefa de desenvolver, mas impacta para melhor visto que não se trata apenas de desenvolver e sim desenvolver com qualidade. O resultado é um produto de alto padrão de qualidade extremamente diferenciado no mercado.