Adailson Almeida1; André Vinicius Vieira Mildemberg1;

*1 Graduando do curso de Análise e desenvolvimento de sistemas do Centro Universitário Unifacear;*

**RESUMO**

*Realizamos o desenvolvimento de um jogo de damas em linguagem Java, para realizar o planejamento do desenvolvimento do sistema utilizamos diagramas UML, como o diagrama de casos de uso e diagrama de classes.*

*O sistema consiste em um jogo de damas com as regras brasileiras do jogo, como por exemplo: A partida é realizada entre dois jogadores, em tabuleiro 8X8 (64 casas) alternadamente brancas e pretas, com 12 peças brancas e 12 peças pretas.*

*Para gerar uma maior competitividade nos jogadores o sistema conterá um ranqueamento entre todos os jogadores com base em pontos que eles ganharão nas vitorias de cada jogo.*

*Palavras chave: damas, jogador, Java, diagramas.*

***ABSTRACT***

*We developed the game of checkers in Java language, to carry out the planning of the development of the system uses UML diagrams, such as the use-case diagram and class diagram.*

*The system consists of a game of checkers with the classic rules of the game, such as: The game is played between two players, on an 8X8 (64) board alternately white and black, with 12 white pieces and 12 black pieces.*

*To generate a greater characteristic in the players the system will contain a ranking among all the players based on points that they will win in the victories of each game.*

*Keywords: checkers, player, Java, diagrams.*

1. **INTRODUÇÃO**

Para este projeto nós desenvolvemos um jogo de damas em linguagem Java com base nos princípios de orientação a objetos (POO) e também foi utilizado o padrão de projeto MVC afim de organizar melhor a estrutura do sistema. Para nos auxiliar no desenvolvimento nos utilizamos a IDE Eclipse.

Além disso, no processo de desenvolvimento nós utilizamos diagramas baseados na UML para definirmos de forma mais concreta o que o sistema deverá ou não conter, para desta forma sermos mais produtivos no desenvolvimento. Os diagramas utilizados foram o diagrama de classes e o diagrama de casos de uso.

1. **DESENVOLVIMENTO**

No tópico de do desenvolvimento abordaremos a variante de regras do jogo de damas que escolhemos no tópico 2.1. No item 2.2 falaremos sobre as ferramentas que utilizamos durante o desenvolvimento do sistema e como nós fizemos uso delas.

Nos itens 2.3 e 2.4 explicaremos os diagramas de casos de uso e de classes respectivamente, os quais foram elaborados com o intuito de facilitar o planejamento do projeto.

Na sequencia no tópico 2.5 contextualizaremos sobre o padrão de projeto model-view-controller que foi utilizado no desenvolvimento do sistema.

Finalmente no item 2.6 apresentaremos os principais algoritmos desenvolvidos para o sistema e uma breve explicação sobre cada um deles.

2.1 REGRAS

De acordo com o site “BrainKing” existem diversas versões variantes no jogo de damas entre elas as damas internacionais, Canadianas, alquerques, checas e brasileiras, que inevitavelmente é a versão mais popular no país. O nosso jogo será baseado na versão brasileira. As regras do jogo de damas brasileiro são:

* Tabuleiro 8X8
* Partida realizada entre dois jogadores
* 24 peças ao todo
* 12 pretas e 12 brancas
* Peças brancas começam
* A peça só anda para frente
* Anda uma casa por vez
* Quando atinge a última linha a peça vira dama
* A dama anda para frente e para trás porém com o limite de uma casa
* A dama não pode “passar por cima” de nenhuma peça aliada
* Se houver a possibilidade a captura é obrigatória
* A peça só pode ser capturada se houver um espaço livre na mesma diagonal que a peça que irá capturar está
* As peças normais podem realizar a captura indo para trás
* Se no mesmo lance se apresentar mais de um modo de capturar, é obrigatório executar o lance que capture o maior número de peças (Lei da Maioria).
* Após 20 lances sucessivos, sem captura ou deslocamento de pedra, a partida é declarada empatada.

2.2 FERRAMENTAS UTILIZADAS NO DESENVOLVIMENTO

Para o desenvolvimento do projeto nos fizemos uso de diversas ferramentas a seguir iremos apresenta-las e falar um pouco sobre sua utilização.

2.2.1 Eclipse

Para a programação do software nós utilizamos o Eclipse que é uma IDE (Integrated Development Environment ou Ambiente de Desenvolvimento Integrado) que focada em Java porém tem suporte para algumas outras linguagens como PHP e Python. Além disso o Eclipse é um software open source.

2.2.2 Draw.io

Criado em setembro de 2016, o Draw.io é um software online grátis usado para criar diagramas. Nele nós elaboremos os diagramas de classe e de casos de uso do projeto.

2.2.3 Biblioteca Swing

Para a criação das interfaces gráficas nós fizemos uso da biblioteca Swing do Java que é uma biblioteca que simplifica o desenvolvimento da parte visual do sistema.

2.3 DIAGRAMA DE CASOS DE USO

Um diagrama de casos de uso é um diagrama dinâmico ou de comportamento na UML (Linguagem de Modelagem Unificada). O cientista de computação JACOBSON descreve casos de uso da seguinte forma, podemos dizer que um caso de uso é um documento narrativo que descreve a sequência de eventos de um ator que usa um sistema para completar um processo.

Com base nesta descrição desenvolvemos o diagrama de casos de uso do projeto tentando incluir todas os casos de uso do projeto.

A seguir na figura 1, o Diagrama de casos de uso do projeto:

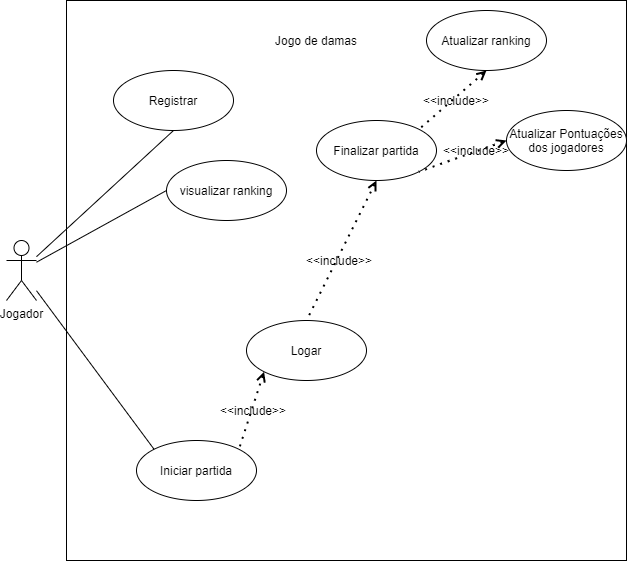


FIGURA 1: Diagrama de casos de uso do projeto.

FONTE: Os autores (2020)

2.4 DIAGRAMA DE CLASSES

Segundo Donald Bell no livro Fundamentos básicos de UML: O propósito do diagrama de classes é mostrar os tipos que estão sendo modelados no sistema e a relações entre eles. Nessa perspectiva observa-se que é fulcral a elaboração de um diagrama de classes para um sistema afim de documentar corretamente o funcionamento do mesmo e facilitar futuras manutenções no código.

Para a melhor visualização do diagrama de classes neste artigo nos dividimos ele em quatro partes que estão representadas nas figuras 2, 3, 4, 5. Após isso dos itens 2.4.1 até o 2.4.X temos uma breve descrição das principais classes do sistema.

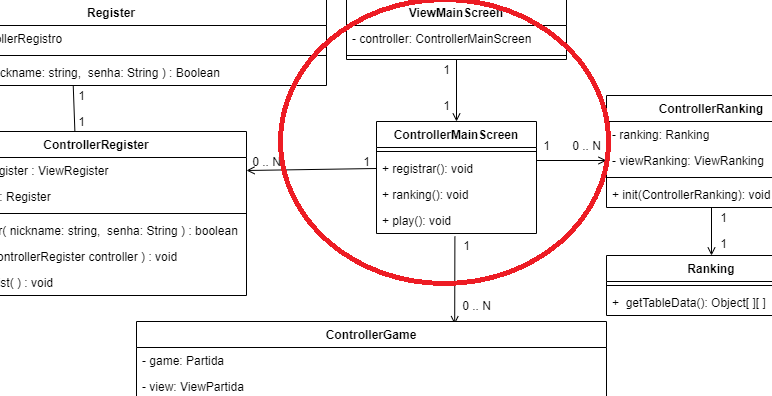


FIGURA 2: Parte central do diagrama.

FONTE: Os autores (2020)

Como nós desenvolvemos nosso sistema baseado nos padrões de projeto MVC, para funcionar como núcleo do sistema temos as classes ViewMainScreen e ControllerMainScreen representadas na figura 2. A partir delas podemos dividir o sistema em três ramificações que são as classes de registro, de ranking e a parte principal do sistema que são as classes da partida.

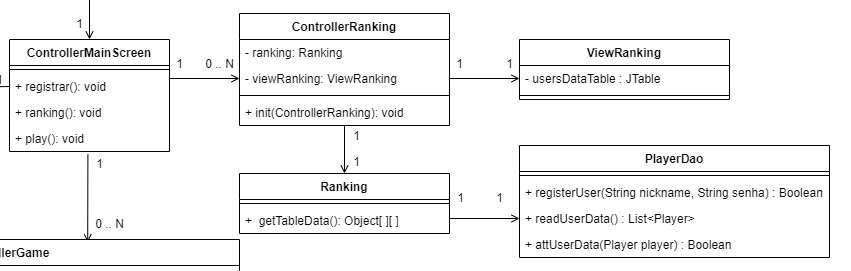


FIGURA 3: Classes de Ranking.

FONTE: Os autores (2020)

Na Ramificação representada na figura 3 do sistema estão as classes Ranking, ViewRanking, ControllerRanking e a classe PlayerDao que é uma classe baseado no padrão de aplicação DAO ( Data Access Object ) ou objeto de acesso a dados ele é utilizado para separar as regras de negócio da aplicação das regras de acesso à base de dados, nesse caso o arquivo TXT, dessa forma deixando o código mais desacoplado e facilitando a reutilização uma vez que seja necessário usar um Banco de dados para aplicação caberia ao desenvolvedor simplesmente criar uma classe que herde PlayerDao e sobrescreva os métodos de acordo com a utilização desse banco.

Essas classes então serão responsáveis por extrair da base de dados do sistema as informações sobre as pontuações de cada jogador e exibir ao usuário ordenando do jogador com maior pontuação para o com menor.

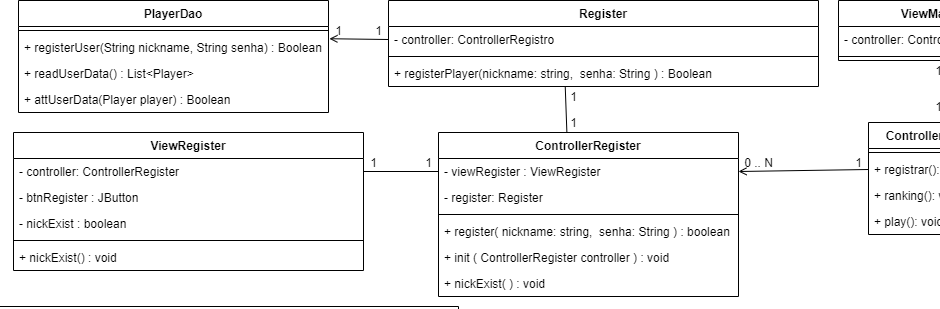


FIGURA 4: Classes de registro.

FONTE: Os autores (2020)

As classes de registro da figura 4 estão separadas nas classes Register, ViewRegister e Controller, além da classe PlayerDao que já foi explicada no texto sobre a figura 3. Essas classes são responsáveis por validar e cadastrar os dados do jogador na base de dados, para ele poder realizar o login posteriormente, para se registrar o jogador precisa informar um nickname e senha além disso o nickname que ele informar não pode já estar registrado no sistema.

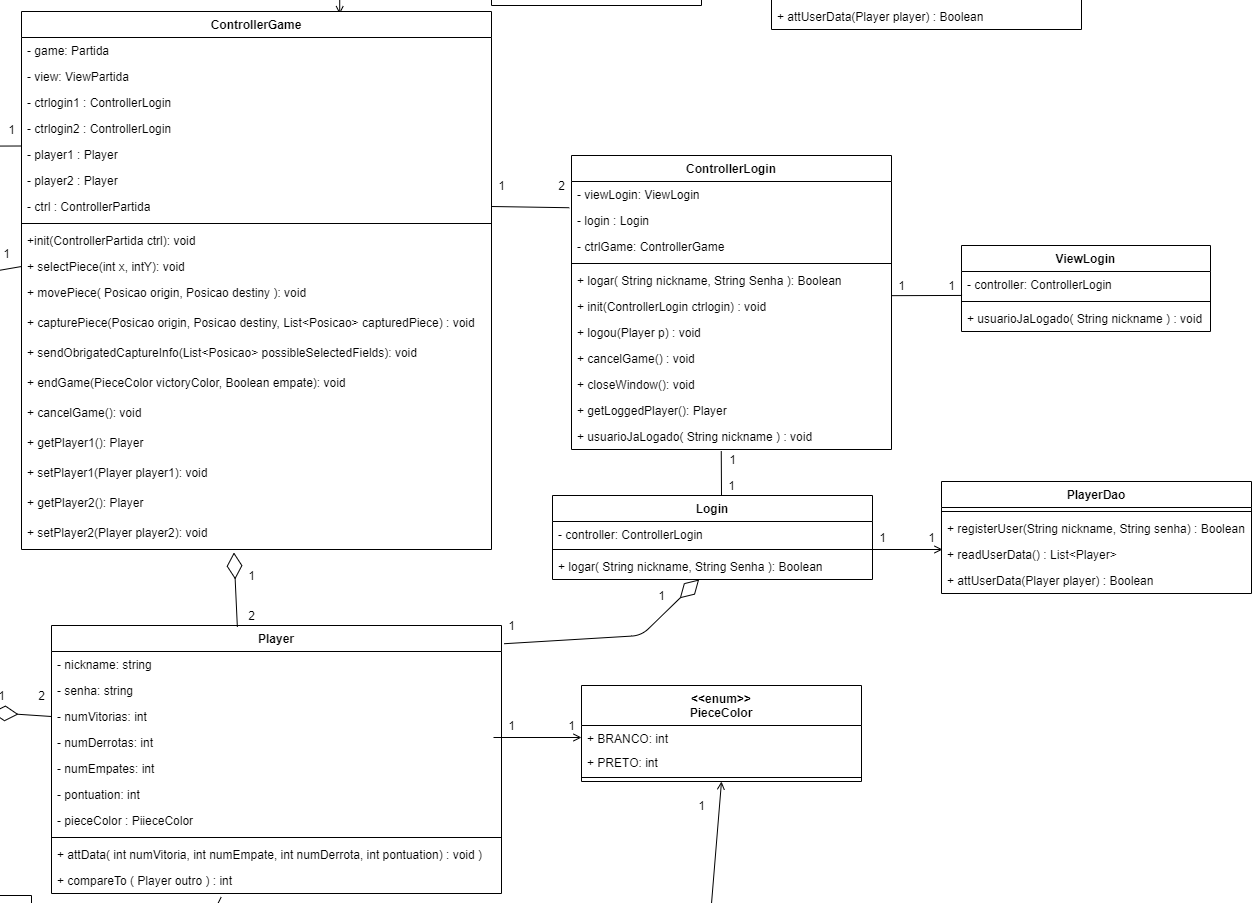


FIGURA 5: Classes de login.

FONTE: Os autores (2020)

Na figura 5 temos as classes utilizadas para realizar o Login dos jogadores antes de iniciar a partida. A partir do momento que a classe ControllerGame é chamada ela automaticamente instancia duas vezes a classe ControllerLogin que por sua vez chama as classes ViewLogin e Login na qual a classe Login também utiliza a classe PlayerDao para acessar a base de dados, então quando o usuário informar dados validos a classe Login instancia a classe Player com os dados desse usuário e passa para a ControllerGame.

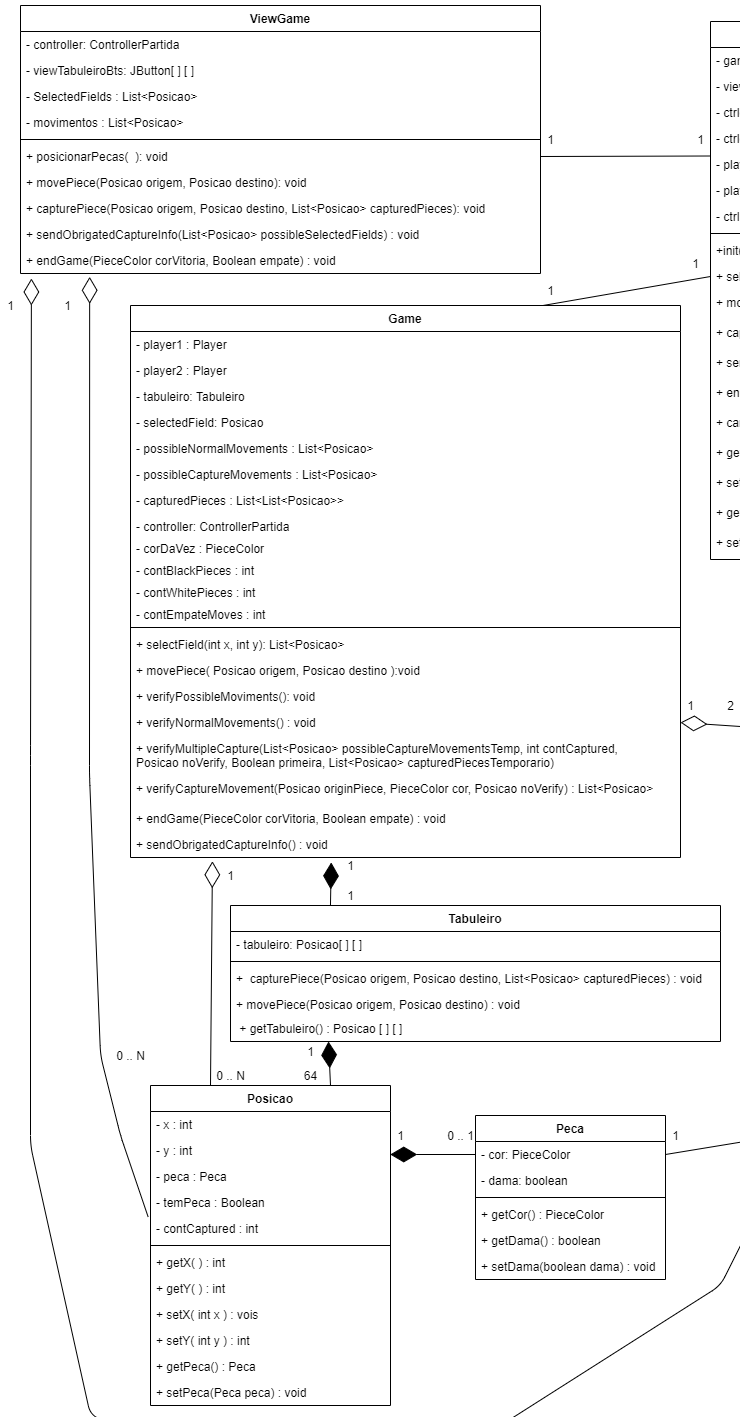


FIGURA 6: Classes da Partida.

FONTE: Os autores (2020)

Na figura 6 temos representadas as classes da Game e ViewGame que estão ligadas a classe ControllerGame representada na figura 5, por questões de uma melhor visualização resolvemos separa-las.

A classe Game da partida contém todas as regras de negócio do jogo, é ela que realiza a validação dos movimentos da partida, organiza a vez de cada jogador, determina o fim da partida, atualiza as estatísticas de cada jogador e salva na base de dados.

A classe tabuleiro é instanciada pela classe Game, ela contém uma matriz de objetos da classe Posicao. Essa classe é utilizada para realizar as validações da classe Game.

A classe Posicao possui os atributos de coordenadas X e Y, um atributo peca que é um objeto da classe Peca e um atributo booleano temPeca para verificar se nessa posição tem uma peça.

A classe Peca tem dois atributos o primeiro contendo um valor da classe enumerada PieceColor para representar a cor dessa peça e um atributo booleano para verificar se a peça é uma dama.

2.5 PADRÃO DE PROJETO

Para a organização do projeto optamos por utilizar o padrão model-view-controller ou simplesmente MVC que segundo Guedes, o princípio básico do MVC é a divisão da aplicação em três camadas: a camada de interação do usuário (**view**), a camada de manipulação dos dados (**model**) e a camada de controle (**controller**).

Na camada view como o próprio nome sugere temos a parte visual do sistema, ela não deve ter conhecimento das regras de negocio do software, sua única responsabilidade deve ser exibir as informações para o usuário capturar seus “inputs” e passar para a camada de controle.

Por sua vez, a camada model contém as regras de negócios da aplicação e o acesso à base de dados quando necessário, sua responsabilidade é receber os dados fazer os tratamentos específicos para eles e enviar para camada de controle não tendo assim nenhuma preocupação com a exibição dos dados.

Finalmente, a camada controller é um mediador entre a view e a model ela que transmite o dado de uma para a outra.

Dessa forma, utilizando o padrão MVC obtemos diversos benefícios como desacoplar a parte visual da parte das regras de negócio, assim facilitando futuras manutenções.

Porém usar o padrão MVC exige certa disciplina dos desenvolvedores uma vez que eles precisam tomar muito cuidado para não podem misturar códigos de uma camada com de outra, além disso usar o MVC em projetos pequenos pode deixá-los mais complexos do que necessitavam.

2.6 ALGORITMOS UTILIZADOS

Segundo Dasgupta, Papadimitriou e Vazirani, "algoritmos são procedimentos precisos, não ambíguos, mecânicos, eficientes e corretos". Em palavras mais coloquiais algoritmos são formulas para resolver problemas.

Na programação de sistemas pode se dizer que qualquer código escrito é um algoritmo dessa forma iremos abordar alguns dos principais algoritmos utilizados no sistema e explicar um pouco seu funcionamento.

**WIP – EM DESENVOLVIMENTO**

1. **CONCLUSÃO**

Dessa forma, através das metodologias e ferramentas apresentadas nesse artigo, eclipse para programação, biblioteca Swing do Java para criação das interfaces gráficas, a elaboração dos diagramas de classes e de casos de uso para ter uma visão melhor do escopo do projeto, o uso do padrão de projeto MVC para deixar o código mais desacoplado e facilitar a reutilização. Além de todos os conceitos de orientação a objetos vistos em aula possibilitaram o desenvolvimento e a finalização do projeto com todas as regras do jogo de damas implementadas além de um sistema de login e registro de usuários com pontuações para cada jogador que podem ser acessadas por um ranking.

1. **REFERÊNCIAS**

### BELL, DONALD. Fundamentos básicos de UML: O diagrama de classes. 19, dezembro de 2016. 1 ed. IBM Corporation.

DASGUPTA**,** Sanjoy**.** PAPADIMITRIOU**.** Christos**,** VAZIRANI**,** Umesh.[**Algorithms**](http://highered.mheducation.com/sites/0073523402/index.html)**.** 13, setembro de 2006. 1 ed. Editora:  McGraw-Hill Higher Education.

### JACOBSON, IVAN: [Object Oriented Software Engineering: A Use Case Driven Approach](https://www.amazon.com.br/Object-Oriented-Software-Engineering-Approach/dp/0201544350/ref=sr_1_3?qid=1585572037&refinements=p_27%3AIvar+Jacobson&s=books&sr=1-3). 1 ed. Editora: Addison Wesley. 1, julho de 1992.

### 

### Regras jogo de damas. 11, maio de 2008. Disponível em: https://brainking.com/pt/GameRules?tp=120 >. Acesso em: 11, de setembro de 2020.

GUEDES, Marylene. 22, junho de 2020. Disponível em: <https://www.treinaweb.com.br/blog/o-que-e-mvc/>. Acesso em: 11, de novembro de 2020.