Projeto de Jogo Scrabble

Especificação de Requisitos de Software

Versão 1.0

23/03/2023

Versão	Autor(es)	Data	Ação
1.0	Eduardo Dani Perottoni, Fernanda Larissa Müller, Lívia Corazza Ferrão	12/03/2023	Definição de requisitos.
1.1	Eduardo Dani Perottoni, Fernanda Larissa Müller, Lívia Corazza Ferrão	29/03/2023	Refatoração dos requisitos a partir da fala com o professor.
2.0	Eduardo Dani Perottoni, Fernanda Larissa Müller, Lívia Corazza Ferrão	12/04/2023	Arrumando conforme as correções apontadas pelo professor na entrega 1.

Conteúdo:

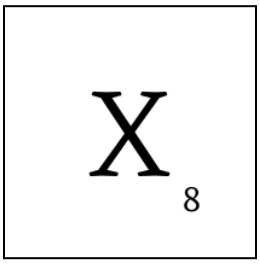
1. Introdução

1.1. Objetivo do desenvolvimento

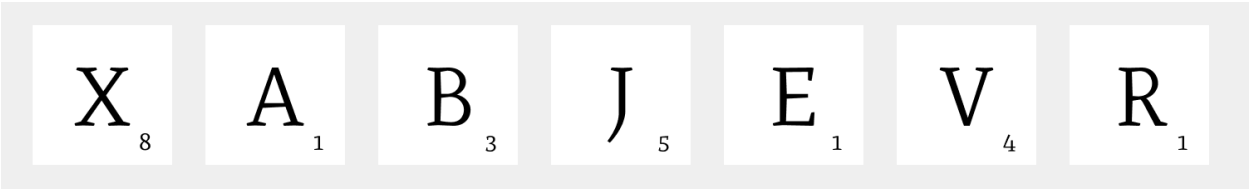
Desenvolvimento de um jogo distribuído que suporte a disputa de partidas de Scrabble na modalidade usuário contra usuário.

1.2. Definições e abreviatura

- ***Card:*** Nesse documento e nos demais, *card* significa cada unidade/carta/letra/peça do jogo. Esta unidade é formada por uma letra do alfabeto e uma pontuação referente a esta carta. Segue uma representação de um *card*:



- ***Pack*****:** Nesse documento e nos demais, *pack* se refere ao conjunto de *cards* pertencentes a algum jogador. Segue abaixo a representação de um *pack*:



- ***Bag*****:** No contexto do jogo, *bag* será o saco de cards, que conterá *cards* reservas. Além disso, quando o jogador não tiver mais cartas no seu *pack*, *cards* da *bag* serão adicionados ao *pack* do jogador. Quando a *bag* não tiver mais peças, não haverá mais troca de *cards*.

Regras do jogo:

- **Número de jogadores:** 2 jogadores.
- **Objetos fundamentais no jogo**
  - *Cards*: explicados acima.
  - Tabuleiro: Tabuleiro 15x15 = 225 posições, com posições normais e posições especiais, que acrescentam valor à pontuação do jogador, quando o mesmo forma uma palavra com algum *card* posto acima delas. As seguintes posições especiais estão presentes no tabuleiro:

        - Double Word = Dobra o valor total da pontuação normal da palavra cuja letra foi inserida na posição.
        - Triple Word = Triplica o valor total da pontuação normal da palavra cuja letra foi inserida na posição.
        - Double Letter = Dobra o valor da letra que foi inserida na posição.
        - Triple Letter = Triplica o valor da letra que foi inserida na posição.
        - Central position = Dobra o valor da palavra.
- **Objetivo do jogo:** Fazer o maior número possível de pontos formando palavras no tabuleiro.
- **Distribuição inicial dos *cards*:** Cada jogador recebe inicialmente 7 *cards* de um "saco de letras" e, ao longo do jogo, em sua vez de jogar, pode trocar quantos *cards* quiser por outros do saco. Porém, se optar pela troca de *cards*, na rodada, deverá abrir mão de construir uma palavra no tabuleiro e acumular pontos.
- **Formação de Palavras:** Na sua vez de jogar, quando não optar pela troca de *cards*, o jogador precisa formar palavras utilizando os *cards* que possui em seu *pack.* As palavras devem ser formadas apenas na horizontal ou na vertical e sempre devem ser formadas a partir de letras que já estão dispostas.
- **Pontuação:** Como já explicado, cada card possui uma pontuação e a pontuação da rodada é dada pela soma dos valores das letras da palavra formada.
- **Passar a vez (troca de turno forçada):** Caso o jogador não consiga formar uma palavra com suas peças, ele pode passar a vez para o outro jogador.
- **Fim do Jogo:** O jogo termina quando todas as letras do saco acabam e um dos jogadores termina com os *cards* do seu *pack* ou se ambos os jogadores passarem o turno duas vezes seguidas. O vencedor é o jogador com mais pontos no final da partida.
- **Regras Especiais:** Há algumas regras especiais, como a restrição de usar apenas palavras do dicionário. Além disso, há uma jogada especial, como um bônus especial de 50 pontos, quando o jogador utiliza todas as suas 7 peças em uma única jogada. A palavra inicial precisa usar a posição central do tabuleiro.
- **Pontuação das letras:**
    - 1 ponto: A, E, I, M, O, R, S, T, U e coringa;
    - 2 pontos: C, D, L e P;
    - 3 pontos: B, Ç e N;
    - 4 pontos: F, G, H e V;
    - 5 pontos: J;
    - 6 pontos: Q;
    - 8 pontos: X e Z;

1.3. Referências

Trie (árvore de recuperação):
https://www.ime.usp.br/~pf/estruturas-de-dados/aulas/tries.html
https://www.geeksforgeeks.org/trie-insert-and-search/

2. Visão Geral do Sistema

2.1. Arquitetura do Software

Programa cliente-servidor distribuído.

2.2. Premissas de desenvolvimento

- O programa deve ser desenvolvido usando a linguagem de programação Python e o paradigma de Orientação a Objetos.
- Especificação de projeto deve ser baseada na segunda versão da linguagem UML.

3. Requisitos de software

3.1. Regras de negócio

RN001 - Posições das letras devem ser conexas no tabuleiro: Os *cards* devem ser inseridos em posições válidas, formando uma palavra apenas na vertical ou na horizontal. Além disso, se houverem palavras existentes já inseridas no tabuleiro, os *cards* inseridos para a formação da palavra, adjacentes à palavra já existente, devem também formar palavras com esta que já estava no tabuleiro.

RN002 - Palavras submetidas devem existir: A palavra deve existir no dicionário da língua portuguesa.

RN003 - Seleção de *card* do *pack*: Em jogadas normais (para construção de palavras), o jogador não pode selecionar mais de um *card* do seu *pack* ao mesmo tempo. Sempre a última seleção é a válida. Isso é exceção apenas para o caso em que o jogador irá trocar seus cards, em que a seleção poderá ser de múltiplas peças.

RN004 - Validação/Invalidação da palavra: Caso a palavra submetida não exista no dicionário, os *cards* usados na tentativa serão recolhidos do tabuleiro e devolvidos ao *pack* do jogador, que deve reiniciar seu turno. Caso a palavra submetida exista, computar-se-á a pontuação do jogador, a jogada será enviada para o outro jogador e será executada a troca de turno, desabilitando o jogador local e o programa ficando no aguardo de jogada do adversário ou de notificação de abandono.

RN005 - Submissão com *cards* selecionados: Se algum *card* do *pack* estiver selecionado no momento da submissão da palavra, o mesmo será ignorado.

RN006 - Troca de cards com a bag: Para a troca de *cards*, o jogador deve escolher quantos *cards* deseja trocar. Esses *cards* serão adicionados à *bag*. Os cards da bag, então, serão selecionados aleatoriamente (os *cards* com valor idêntico aos *cards* selecionados para troca não serão considerados) para troca e inseridos ao *pack*.

RN007 - Estrutura de envio/recebimento da jogada: A estrutura da jogada enviada deve conter o tipo da jogada feita pelo jogador local (pode ser uma jogada de troca, de construção de palavra ou de passar a vez). Caso seja uma jogada de construção, deve-se informar as posições ocupadas no tabuleiro e os *cards* que compõem a palavra recém construída, assim como a pontuação feita pelo jogador.

RN008 - Término da partida: Existem duas possibilidades para o fim de uma partida. O primeiro caso acontece quando os dois jogadores desistem de um *round* duas vezes seguidas (4 desistências seguidas). O outro caso acontece quando todos os *cards* da *bag* forem utilizados e um dos jogadores terminar os *cards* do seu *pack*. No primeiro caso, os *cards* restantes no(s) *pack(s)* do(s) jogador(es) são ignorados.

RN009 - Primeira submissão: Na primeira jogada o jogador deve adicionar no mínimo dois *cards* ao tabuleiro e um dos *cards* adicionados deve estar na posição central do tabuleiro, ou seja, na posição 7x7 (considerando a primeira posição sendo 0 e a última 14) do tabuleiro.

RN010 - Jogada especial: O jogador irá receber um bônus de 50 pontos caso forme uma palavra válida utilizando todos os 7 *cards* do seu *pack* em um *round*.

3.2. Requisitos Funcionais

RF001 - Inicializar o programa: Quando executado, o programa deve apresentar uma interface mostrando o tabuleiro do jogo, com as 7 peças do jogador e solicitar o nome do participante. Em seguida, deve solicitar conexão com DOG Server por meio dos recursos de DOG. O resultado da tentativa de conexão deve ser informado ao usuário. Na conexão bem sucedida as demais funcionalidades serão habilitadas. No caso de conexão mal sucedida, o programa se encerra.

RF002 - Iniciar o jogo: O programa possui uma opção de iniciar o jogo em seu menu. Quando acionado, será enviado uma solicitação de início a Dog Server, o qual irá definir a ordem dos jogadores e identificá-los. Caso não atinja o número mínimo de participantes, a partida não inicia e o programa é encerrado. A interface é habilitada para o jogador da vez com o tabuleiro em seu estado inicial.

RF003 - Reiniciar partida: O menu possui a opção de reiniciar a partida, o qual leva o programa para seu estado inicial, ou seja, sem partida em andamento e tabuleiro sem *cards*. Os jogadores perdem seus *cards* e novos são inseridos em seu *pack* para início de uma nova partida. Esta funcionalidade só deve estar habilitada ao término do jogo.

RF004 - Selecionar *card* do *pack*: O jogador habilitado deve selecionar um *card* do seu *pack*. A peça selecionada será destacada das demais, obedecendo a RN003.

RF005 - Selecionar posição do tabuleiro: A posição de destino deve ser habilitada após o jogador selecionar o *card* de seu *pack*. O programa avaliará se a posição de destino está vaga para receber a peça selecionada. Caso a jogada não for válida, ou

seja, a peça não poder ocupar a posição de destino, ela não será movida, mas continuará selecionada. No caso de êxito na colocação da letra, em destino válido, o *card* será movido do *pack* para a posição do tabuleiro.

RF006 - Submeter palavra: Após a colocação dos *cards* desejados no tabuleiro, para a formação da palavra, sendo todas as seleções válidas, o jogador deve clicar na opção de “Enviar palavra”. Essa ação é necessária para que o programa cumpra a RN002, não permitindo palavras inválidas. Para validação da palavra, o programa fará uma busca em uma estrutura de dados, conferindo se a palavra existe e seguindo a RN004, RN001 e RN005. A primeira submissão do primeiro jogador deve seguir a RN009. Caso o jogador utilize os 7 *cards* do seu *pack* irá receber pontuação bônus conforme a RN010. No caso de encerramento de partida, deve ser notificado o nome do jogador vencedor (obedecendo a RN008).

RF007 - Trocar letras do *pack*: Quando o jogador estiver com dificuldade em encontrar uma palavra com os *cards* que possui no seu *pack*, ele pode trocar algum destes ou todos por outros *cards* aleatórios do saco, obedecendo a RN006. Quando o jogador utilizar essa opção, acontece a troca de turno, ou seja, não é possível submeter uma palavra. Este tipo de jogada pode ser utilizada quantas vezes for desejado pelo jogador, até que existam *cards* na *bag*.

RF008: Desistir da rodada: Caso o jogador julgar estar sem opções, pode apenas solicitar a desistência da jogada, essa opção realiza a troca de turno. Quando esta ação for acionada, deve-se checar o fim da partida, obedecendo a RN008.

RF009 - Receber determinação de início: Programa deve receber uma notificação de início de jogo, originária do DOG Server, em função da solicitação de início de jogo por parte de outro jogador. A partir do recebimento da notificação de início, é o descrito em RF002. Ou seja, a interface do programa deve ser atualizada com as informações recebidas na determinação.

RF010 - Receber jogada: Procedimento que recebe jogada do adversário, recebida pelo DOG Server. A jogada recebida deve ser um lance regular e obedecer a RN007. Em caso de ser uma jogada de construção, a interface deve ser atualizada com as informações recebidas.

RF011 - Receber notificação de abandono: O programa deve ser capaz de receber notificação de abandono da partida do usuário remoto, via DOG Server. Neste caso, a partida se encerra e o abandono deve ser notificado ao usuário local via interface.

RF012 - Retornar *cards* do tabuleiro ao *pack:* Após adicionar *cards* no tabuleiro, no mesmo *round* e antes de submeter a palavra, o jogador pode retornar os mesmos para seu *pack*, para reiniciar a colocação dos mesmos.

3.3. Requisitos não Funcionais

RNF001 - Performance da busca de palavras: A busca para validação de palavras deve ser implementada utilizando uma estrutura de dados que forneça menor complexidade possível de busca, tornando-a mais rápida. Uma possível implementação pode ser utilizando uma Trie (ver seção de referências).

RNF002 - Suporte à execução distribuída: O framework DOG deve ser usado para suportar execução distribuída.

RNF003 - Suporte à Especificação de Projeto: Toda a especificação de projeto deve ser produzida pela ferramenta Visual Paradigm.

RNF004 - Implementação da Interface: A interface do programa deve ser implementada usando a ferramenta TKinter, obedecendo as figuras a seguir.