

Projeto Pontu

Especificação de Requisitos de Software

Versão 1.0

07/09/2022

| Versão | Autor(es) | Data | Ação |
|--------|--|------------|------------------------------------|
| 1.0 | João Pedro Lopes de Camargo Rafael Francisco Réus Stephan Krug | 07/09/2022 | Estabelecimento dos requisitos |
| 2.0 | João Pedro Lopes de Camargo Rafael Francisco Réus Stephan Krug | 16/10/2022 | Detalhamento das regras do jogo |

Conteúdo:

1. Introdução;
 2. Visão geral de Sistema;
 3. Requisitos de Software.
- Apêndice: Regras do Pontu

1. Introdução

1.1 Objetivo

Desenvolvimento de um programa distribuído que suporte a disputa de partidas de Pontu na modalidade usuário contra usuário.

1.2 Definições, abreviaturas

Regras do jogo: ver apêndice.

1.3 Referências:

Apresentação das regras originais (o jogo que será implementado consiste em uma adaptação, conforme apêndice) e exemplo do jogo (vídeo do canal VemKaJogar):

 CONHEÇA PONTU - UM JOGO DE TABULEIRO PARA TODA A FAMÍLIA!

2. Visão Geral

2.1 Arquitetura do programa

Cliente-servidor distribuído.

2.2 Premissas de desenvolvimento

- O programa deve ser implementado em Python;
- O programa deve usar DOG como suporte para execução distribuída;
- Além do código, deve ser produzida especificação de projeto baseada em UML, segunda versão.

3. Requisitos de Software

3.1 Requisitos Funcionais:

Requisito funcional 1 – Iniciar programa: Ao ser executado, o programa deve apresentar na interface o tabuleiro do jogo em seu estado inicial, com o tabuleiro vazio. Após isso, deve solicitar conexão com o DOG Server (utilizando os recursos de DOG). O resultado da tentativa de conexão deve ser informado ao usuário. Apenas em caso de conexão bem sucedida as demais funcionalidades estarão habilitadas. No caso de conexão mal sucedida, as alternativas são encerrar o programa e tentar reconectar.

Requisito funcional 2 – Iniciar jogo: O programa deve apresentar a opção de menu “*iniciar jogo*” para o início de uma nova partida. O procedimento de início de partida consiste em enviar uma solicitação de início a Dog Server, que retornará o resultado, que será a identificação e a ordem dos jogadores, em caso de êxito, ou a razão da impossibilidade de início de partida, caso contrário. A interface do programa deve ser atualizada com as informações recebidas e caso o jogador local seja quem inicia a partida, a interface deve estar habilitada para seu procedimento de lance. Esta funcionalidade só deve estar habilitada se o programa estiver em seu estado inicial, isto é, sem partida em andamento e com o tabuleiro em seu estado inicial;

Requisito funcional 3 – Restaurar estado inicial: O programa deve apresentar a opção de menu “*restaurar estado inicial*” para levar o programa ao seu estado inicial, isto é, sem partida em andamento e com o tabuleiro em seu estado inicial. Esta funcionalidade só deve estar habilitada se o programa estiver com uma partida finalizada;

Requisito funcional 4 – Selecionar origem: O programa deve permitir a um jogador habilitado selecionar uma peça sua presente em uma posição do tabuleiro. A peça selecionada deve ser visualmente destacada da interface do programa. Esta ação só deve estar habilitada quando nenhum jogador tiver peças a serem posicionadas e o jogador em questão puder mover a peça que tentou selecionar, conforme as regras do jogo (ver apêndice). Se a ação for executada após o jogador já ter selecionado origem ou não puder selecionar nenhuma peça, deve ser notificado lance irregular e o programa deve novamente aguardar a primeira ação do jogador habilitado (selecionar origem ou selecionar destino);

Requisito funcional 5 – Selecionar destino: O programa deve permitir a um jogador habilitado selecionar uma posição do tabuleiro, onde será colocada uma peça previamente selecionada de uma posição do tabuleiro ou das suas peças ainda não posicionadas - de acordo com as regras do jogo (ver apêndice). Esta funcionalidade só deve estar habilitada se o programa tiver peças ainda fora do tabuleiro ou com posição origem selecionada (ver Requisito funcional 4). O programa deve avaliar se a posição destino pode receber a peça selecionada, de acordo com as regras do jogo (ver apêndice). Em caso positivo, deve efetuar a jogada - isto é, remover a peça de seu posicionamento inicial e colocá-la na posição destino - e, em caso negativo, notificar o lance irregular e o programa deve novamente aguardar a primeira ação do jogador habilitado (selecionar origem). No caso de êxito na colocação de peça em seu destino, o programa deve permitir que o jogador retire uma ponte (ver Requisito funcional 6) ou enviar a jogada ao adversário (utilizando os recursos de DOG) - caso ainda haja peças na a serem colocadas por parte de algum dos jogadores. A jogada enviada deve conter a posição de origem do tabuleiro (no caso de seleção de origem) e a posição destino. Se ainda houver peças a serem posicionadas por algum dos jogadores, o jogador local deve ser desabilitado e o programa fica no aguardo de jogada do adversário (ver Requisito funcional 9) ou de notificação de abandono (ver Requisito funcional 10);

Requisito funcional 6 – Retirar ponte: O programa deve permitir a um jogador habilitado selecionar uma ponte qualquer do tabuleiro para ser removida. Esta funcionalidade só deve estar habilitada se o jogador tiver terminado de selecionar destino (ver Requisito funcional 5) ou não puder mover nenhuma peça, conforme as regras do jogo (ver apêndice). O programa deve avaliar se a ponte selecionada está no tabuleiro. Em caso

positivo, deve removê-la. Em caso negativo, deve notificar o lance irregular e deve aguardar novamente a ação do jogador habilitado. No caso de êxito na remoção da ponte, o programa deve verificar se uma peça deve ser removida do tabuleiro (ver Requisito funcional 7).

Requisito funcional 7 – Verificar remoção de peça: O programa deve poder verificar se uma peça no tabuleiro foi isolada, de acordo com as regras do jogo (ver apêndice). Esta funcionalidade só deve ser habilitada quando uma ponte for removida do tabuleiro. Depois disso, o programa deve enviar a jogada ao adversário (utilizando os recursos de DOG) e avaliar o encerramento de partida. No caso de encerramento de partida, deve ser notificado se o jogador foi vencedor ou perdedor; no caso de não encerramento, deve ser desabilitado o jogador local e o programa fica no aguardo de uma jogada do adversário (ver Requisito funcional 9) ou de uma notificação de abandono (ver Requisito funcional 10).

Requisito funcional 8 – Receber determinação de início: o programa deve ser capaz de receber uma notificação de início de partida, originada em Dog Server, em função de uma requisição de início de partida por parte de um outro jogador conectado ao servidor. As etapas a partir do recebimento da notificação de início são as mesmas descritas no 'Requisito funcional 2 – Iniciar jogo', isto é, a interface do programa deve ser atualizada com as informações recebidas e caso o jogador local seja quem inicia a partida, a interface deve estar habilitada para seu procedimento de lance, conforme as regras do jogo (ver apêndice).

Requisito funcional 9 – Receber jogada: o programa deve poder receber uma jogada do adversário, enviada pelo Dog Server, quando for o turno do adversário do jogador local. A jogada recebida deve ser um lance regular e conter as informações especificadas para o envio de jogada no 'Requisito funcional 5 – Selecionar destino', no 'Requisito funcional 6 - Retirar ponte' e no 'Requisito funcional 7 - Verificar remoção de peça'. O programa deve então remover a peça da origem definida e colocá-la no destino, bem como remover a ponte do tabuleiro e, caso de acordo com as regras do jogo (ver apêndice), remover uma peça. Caso o adversário tenha apenas capacidade de remover uma ponte, o programa deve apenas remover a ponte selecionada e, caso de acordo com as regras do jogo (ver apêndice), remover uma peça. Após o recebimento da jogada, o programa deve avaliar se a partida deve ser encerrada. No caso de encerramento de partida, deve ser notificado o se o jogador em questão foi vencedor ou perdedor; no caso de não encerramento, deve ser habilitado o jogador local, para que possa realizar seu lance;

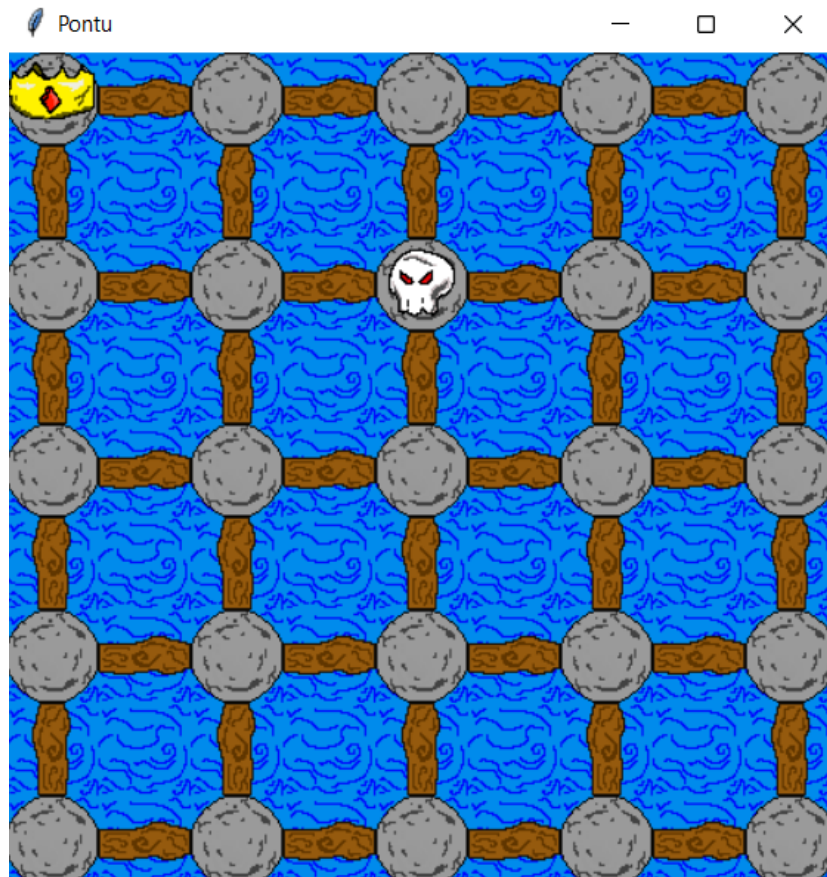
Requisito funcional 10 – Receber notificação de abandono: o programa deve ser capaz de receber uma notificação de abandono de partida pelo adversário remoto, enviada por meio do Dog Server. A partida será considerada encerrada e o abandono notificado na interface.

3.2 Requisitos Não Funcionais

Requisito não funcional 1 – Tecnologia de interface gráfica para usuário: A interface gráfica deve ser baseada em *TKinter*;

Requisito não funcional 2 – Suporte para a especificação de projeto: a especificação de projeto deve ser produzida com a ferramenta Visual Paradigm;

Requisito não funcional 3 – Interface do programa: A interface do programa será produzida baseando-se no esboço da imagem abaixo;



Apêndice: Regras do Pontu

O jogo Pontu é disputado entre 2 jogadores em um tabuleiro de 25 posições, organizado como uma matriz 5x5 e contendo 40 caminhos ortogonais - chamados de pontes - entre essas posições. O objetivo de cada jogador é eliminar as peças do adversário.

Elementos do jogo

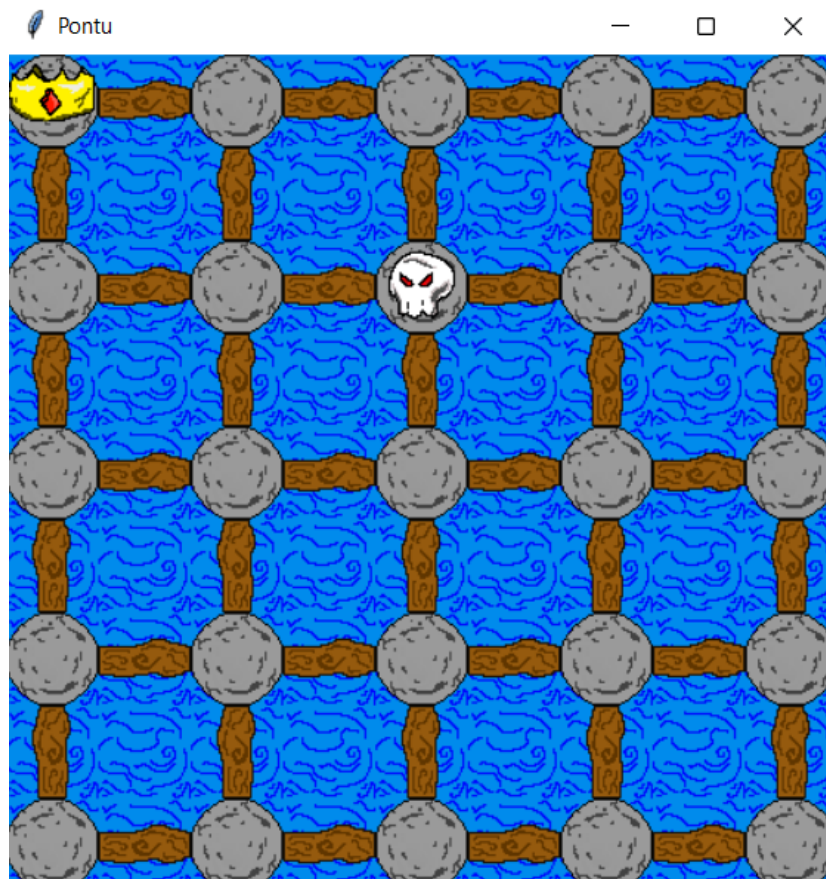


Figura 1 - elementos do jogo Pontu

Além do tabuleiro 5x5, representado na área do tabuleiro na figura 1, o jogo conta com 40 pontes ortogonais entre essas posições, além de 6 peças para cada jogador. Cada jogador inicia a partida com 6 peças, de figura diferente das peças do adversário.

Lances dos jogadores

Inicialmente os jogadores devem alternadamente (adotando algum critério para definir o jogador que inicia a partida) posicionar suas peças em uma das posições não ocupadas do tabuleiro - não sendo possível posioná-las sobre pontes. Depois de todas as peças terem sido posicionadas no tabuleiro, o jogador deve efetuar os seguintes lances: mover uma de suas peças e, em seguida, retirar uma ponte. Caso o jogador não tenha a capacidade de mover nenhuma de suas peças, mesmo com pontes em torno de sua posição, isto é, há outras peças bloqueando seu movimento, ele deve somente retirar uma ponte do tabuleiro e passar sua vez.

Mover uma de suas peças: consiste em selecionar uma de suas peças e com ela mover a uma posição vazia adjacente que esteja ligada por uma ponte à posição da peça antes de iniciado o lance.

Retirar uma ponte: após mover uma de suas peças, o jogador deve retirar uma ponte qualquer do tabuleiro. A cada ponte retirada, deve-se verificar se uma peça foi isolada e retirá-la, ou seja, caso uma peça não tenha mais opção de movimentação, por não haver mais pontes adjacentes a ela, ela deve ser removida do tabuleiro. Caso o jogador escolha remover uma ponte que leve ao isolamento de uma peça sua, essa jogada deve ser permitida e, se essa for a última peça do jogador, essa jogada resulta em sua derrota, mesmo se também causar o isolamento da última peça do adversário.

Encerramento da partida

A partida se encerra apenas quando um jogador não tiver mais peças no tabuleiro - com atenção ao caso especial descrito em “Retirar uma ponte” - sendo o vencedor aquele que ainda possuir peças no tabuleiro.