

**UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE SISTEMAS DA INFORMAÇÃO**

LUIS FELIPE DE AZAMBUJA FEYH
NICOLAS LAZZERI PIMENTA
PEDRO HENRIQUE GOMES MAGRI

Especificação de Requisitos do jogo Salto do Cavalo

FLORIANÓPOLIS
2024

Índice

Índice.....	2
1. Introdução.....	3
1.1 Objetivo.....	3
1.2 Definições e abreviaturas.....	3
1.3 Referências.....	3
2. Visão geral do sistema.....	3
2.1 Arquitetura do programa.....	3
2.2 Premissas de desenvolvimento.....	3
3. Requisitos de aplicação.....	3
3.1 Requisitos funcionais.....	3
3.2 Requisitos não funcionais.....	4
4. Apêndice.....	5
4.1 Objetivo.....	5
4.2 Elementos do jogo.....	5
4.3 Regras do jogo.....	5
4.3.1 Lances do jogador.....	5
4.3.2 Término do jogo.....	6

Versão	Autores	Data	Ação
1.0.0	LUIS FELIPE DE AZAMBUJA FEYH NICOLAS LAZZERI PIMENTA PEDRO HENRIQUE GOMES MAGRI	23/09/2024	Desenvolvimento inicial dos requisitos

1. Introdução

1.1 Objetivo

Desenvolver um programa distribuído para o jogo de tabuleiro "Salto do Cavalo", que possibilite a participação simultânea de dois jogadores.

1.2 Definições e abreviaturas

- **Regras do jogo:** Disponíveis no apêndice
- **RF:** Requisito Funcional
- **RNF:** Requisito Não Funcional

1.3 Referências

Apresentação das regras do jogo e simulação de partida (vídeo do canal Vem Ka Jogar):

<https://www.youtube.com/watch?v=coi-QatOE8I>

2. Visão geral do sistema

2.1 Arquitetura do programa

Programa orientado a objetos, cliente-servidor distribuído.

2.2 Premissas de desenvolvimento

- O programa deve ser implementado em *Python*;
- O programa deve usar o framework *DOG* como suporte para execução distribuída;
- Deve haver uma especificação de projeto baseada em UML2.

3. Requisitos de aplicação

3.1 Requisitos funcionais

- **RF1 | Inicializar:** Quando o programa for executado, ele deve solicitar o nome do jogador e deve tentar estabelecer uma conexão com o servidor DOG. Em seguida, o resultado dessa tentativa de conexão deve ser informado ao usuário. Somente em caso de sucesso na conexão, as demais funcionalidades do jogo serão habilitadas. Caso a conexão falhar, a única opção disponível será encerrar o programa.

- **RF2 | Iniciar partida:** O programa deve exibir a opção de menu "Iniciar jogo" para o início de uma nova partida. Ao selecionar essa opção, será enviada uma solicitação de início ao servidor DOG, que retornará:
 - O resultado contendo a identificação e a ordem dos jogadores, em caso de sucesso.
 - O motivo da impossibilidade de iniciar a partida, em caso de falha.

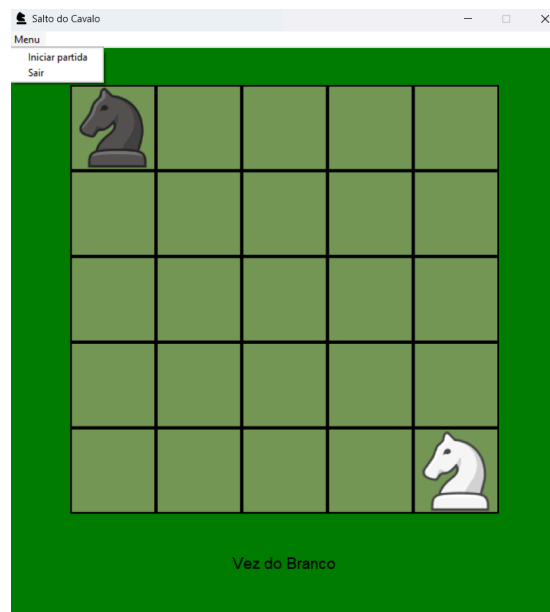
A interface do programa deverá ser atualizada com as informações recebidas, e, se o jogador local for o primeiro a jogar, a interface deverá estar habilitada para o seu movimento. Essa funcionalidade só deve estar disponível se o programa estiver em seu estado inicial, ou seja, sem uma partida em andamento.

- **RF3 | Selecionar posição / Fazer jogada:** O programa deve permitir que um jogador habilitado selecione uma posição válida conforme as regras do jogo. Em seguida, o programa deverá avaliar se a posição escolhida pode receber a peça, se for possível, a peça deve ser movida de sua posição inicial para a posição de destino, enviando também a jogada ao jogador remoto através do DOG Server. Após isso, o programa deve verificar se houve um vencedor na partida. Se a partida terminar, uma notificação na tela deve indicar o nome do jogador vencedor, caso contrário, o turno deve passar do jogador local para o jogador remoto.
- **RF4 | Receber início de partida:** O programa deve ser capaz de receber uma notificação de início de partida do DOG Server, a qual é emitida pelo jogador remoto que já está conectado ao servidor ao iniciar uma partida. Após receber a notificação, as funcionalidades de RF2 deverão ser executadas.
- **RF5 | Receber jogada:** O programa deve ser capaz de receber a jogada do adversário, enviada pelo servidor DOG, quando for a vez do oponente do jogador local. A jogada recebida deve seguir o formato de um lance válido e conter as informações necessárias para o envio de jogadas. Em seguida, o programa deve mover a peça do adversário para o destino indicado e deve-se verificar se a partida chegou ao fim. Caso a partida tenha terminado, uma notificação na tela deve informar o nome do jogador vencedor; caso contrário, o turno passa do jogador remoto para o jogador local.
- **RF6 | Receber notificação de abandono:** O programa deve ser capaz de receber uma notificação de abandono do jogador remoto, enviada pelo servidor DOG. Ao receber essa notificação, a partida deve ser encerrada, e a interface deve informar o abandono, declarando o jogador local como o vencedor.

3.2 Requisitos não funcionais

- **RNF1 | Tecnologia de interface gráfica:** A interface gráfica deve ser desenvolvida com o framework Tkinter.

- **RNF2 | Especificação de projeto:** A modelagem de diagramas UML deve ser realizada com o software Visual Paradigm.
- **RNF3 | Interface gráfica:** A aplicação deve possuir uma interface gráfica semelhante ao esboço fornecido abaixo.



4. Apêndice

4.1 Objetivo

O jogo "Salto do Cavalo" é uma competição estratégica entre dois jogadores em um tabuleiro de matriz 5x5, onde cada um controla uma peça que se movimenta conforme o padrão do cavalo no xadrez. Após cada movimento, a posição ocupada anteriormente pela peça se torna inutilizável, criando áreas bloqueadas no tabuleiro. O objetivo do jogo é forçar o oponente a ficar sem movimentos possíveis, de modo que o jogo termina quando um dos jogadores não consegue mover sua peça para uma posição livre.

4.2 Elementos do jogo

- Tabuleiro de matriz 5x5;
- 1 cavalo para cada jogador.

4.3 Regras do jogo

4.3.1 Lances do jogador

O movimento da peça do cavalo é dado por uma casa em uma direção (horizontal ou vertical), seguidas de duas casas na direção perpendicular. Além disso, o lance deve

atender duas condições para ser válido: estar dentro dos limites do tabuleiro e a posição escolhida não ter sido visitada antes (posição bloqueada).

4.3.2 Término do jogo

O jogo termina quando o jogador habilitado não é capaz de realizar um movimento válido. Vence o oponente que fez o movimento válido anteriormente.