Projeto Ataxx

Especificação de Requisitos de Software

Versão 2.0

22/06/2025

Versão	Autores	Data	Ação
1.0	Antônio Torres, Bianca Verzola e Lucas Brand	09/04/2025	Estabelecimento dos requisitos
2.0	Antônio Torres, Bianca Verzola e Lucas Brand	22/06/2025	Inclusão da lógica de verificação de jogadas possíveis antes da troca de turno; atualização da condição de encerramento de partida

Conteúdo:

1. Introdução:

1.1 Objetivo:

Desenvolvimento de um programa distribuído que permite a disputa de partidas do jogo de tabuleiro Ataxx, de forma que um usuário possa jogar com outro de forma remota.

1.2 Definições, abreviaturas:

As definições e abreviaturas estão descritas no apêndice referente às regras do jogo.

1.3 Referências:

Simulador online para jogo: https://www.onlinesologames.com/ataxx
Detalhes sobre jogo: https://en.wikipedia.org/wiki/Ataxx

2. Visão geral:

2.1 Arquitetura do programa:

Cliente-Servidor distribuído.

2.2 Premissas de desenvolvimento:

O programa em questão deve:

- Ser implementado em Python;
- Usar DOG como framework de suporte para execução distribuída;
- Além do código, ter uma especificação de projeto baseado na segunda versão do UML.

3. Requisitos de Software:

3.1 Requisitos funcionais:

Requisito funcional 1 - Iniciar programa: ao ser executado, o programa deve exibir na interface o tabuleiro do jogo (7x7) em seu estado inicial (ou seja, quatro peças distribuídas, uma em cada um dos cantos do tabuleiro, sendo duas de cada jogador). Na parte superior

da tela, deve ser exibido o logotipo do jogo, enquanto na parte inferior uma caixa de texto deve apresentar o texto “Bem-vindo, esperando conexão”. O programa, em seguida, solicita ao usuário que informe seu nome e, após essa entrada, inicia automaticamente o processo de conexão com o DOG Server (utilizando os recursos de DOG). O resultado da tentativa de conexão deve ser informado ao usuário. Caso a conexão seja estabelecida com sucesso, a mensagem na caixa de texto deve ser atualizada para “Esperando início da partida” e as funcionalidades de iniciar a partida e receber a determinação de início devem ser habilitadas. Caso contrário, a única alternativa deve ser encerrar o programa.

Requisito funcional 2 - Iniciar partida: assim que o usuário estiver conectado ao DOG Server, o menu “iniciar partida” deve ser ativado. Assim que acionado, acontece primeiramente o envio de uma solicitação de início ao DOG Server, que retornará como resultado de êxito, a identificação dos jogadores e a ordem dos jogadores, ou como resultado de falha, a razão que impossibilitou o início da partida. A interface do programa deve ser atualizada, exibindo, no painel de texto, qual o jogador que procede o turno e, no canto superior direito da tela, um disco colorido (amarelo ou vermelho) indicando a cor do jogador local. Além disso, se o jogador local que detém o turno, sua interface deve permitir que ele selecione alguma peça para movimentar. Essa funcionalidade só pode ser habilitada em duas ocasiões. A primeira delas, assim que o requisito funcional 1 é realizado, e a segunda, quando o requisito funcional 3 é ativado.

Requisito funcional 3 - Restaurar estado inicial: o programa deve apresentar em seu menu a opção “Restaurar estado inicial” para levar o programa ao seu estado de início, isto é, o tabuleiro reiniciado e sem partida em andamento. Essa opção só deve ser habilitada ao fim de uma partida.

Requisito funcional 4 - Selecionar peça: Caso o jogador esteja em seu turno, o programa deve permitir que ele selecione uma de suas peças distribuídas no tabuleiro. A peça selecionada deve ser visualmente destacada no tabuleiro, e todas as possíveis posições para as quais ela pode ser movida também (de formas diferentes). Se a ação for realizada após o jogador já ter selecionado uma peça, teremos apenas a alteração da peça em destaque para a última que foi selecionada. Nas situações em que o usuário tenta selecionar uma peça do adversário ou um espaço em branco, o programa apresenta a mensagem “Jogada Inválida!”.

Requisito funcional 5 - Selecionar destino: Com a peça já selecionada (a partir do ‘Requisito funcional 4 - Selecionar peça’), o usuário habilitado tem as opções de possíveis destinos já indicadas na tela. Então, o usuário deve selecionar uma possível nova posição para sua peça. Novamente, caso o usuário selecione uma posição não possível, o programa apresenta a mensagem “Jogada Inválida!”, indicando que o movimento é inválido. O programa deve avaliar qual é o tipo de movimento que será executado dependendo da posição destino, seguindo sempre as regras do jogo (ver no apêndice). Caso a nova posição esteja a apenas uma casa de adjacência da peça selecionada, então a peça é duplicada para a nova posição. Caso a posição de destino esteja duas casas de adjacência distante da posição de origem, então a peça é somente movimentada para essa. Além disso, considerando a posição final da peça movimentada, todas as peças do jogador adversário que sejam vizinhas da peça recém movida trocam de cor, se tornando peças do jogador local também.

No caso de êxito do movimento (seja ele deslocamento da peça ou duplicação), o programa deve enviar a jogada ao adversário (utilizando os recursos do DOG) e então avaliar o encerramento da partida. O envio da jogada deve conter a posição (linha e coluna) da peça selecionada e a posição (linha e coluna) do destino. Caso a partida seja encerrada, deve ser notificado o usuário vencedor. No caso de não encerramento, o programa deve verificar se o jogador adversário possui ao menos um movimento possível para realizar caso ele seja habilitado. Se houver algum movimento disponível, o jogador local deve ser desabilitado e o programa fica então no aguardo do jogador adversário (‘Requisito funcional 7 - Receber jogada’) ou de uma notificação de abandono (‘Requisito funcional 8 - Receber notificação de abandono’). Caso o jogador adversário não tenha nenhum movimento possível, o jogador local permanece habilitado e pode selecionar uma nova peça para jogar novamente (‘Requisito funcional 4 - Selecionar peça’).

Requisito funcional 6 - Receber determinação de início: O programa deve poder receber uma notificação de início de partida, originada em DOG server, em função da solicitação de início de partida por parte do outro jogador conectado no servidor. O procedimento a partir do recebimento da notificação de início é o mesmo descrito no ‘Requisito funcional 2 - Iniciar partida’, 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 já deve estar habilitada para seu procedimento de lance.

Requisito funcional 7 - Receber Jogada: O programa deve poder receber a jogada realizada pelo adversário, enviada pelo DOG Server, quando a vez do jogador adversário ao local for terminada. A jogada recebida deve ser um lance regular e conter informações especificadas para o envio da jogada no ‘Requisito funcional 5 - Selecionar destino’. O programa deve remontar o tabuleiro, de forma que ele corresponda ao estado atual da partida. Após isso, o programa deve avaliar se a partida foi encerrada. Em caso positivo, o jogador vencedor deve ser exibido. Caso contrário, verifica-se se o jogador local possui ao menos um movimento possível para realizar se ele for habilitado. Se tiver, ele é habilitado para que possa proceder seu lance; do contrário, permanece desabilitado.

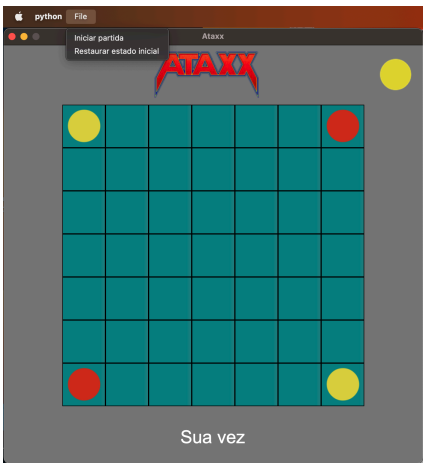
Requisito funcional 8 - Receber notificação de abandono: o programa deve poder receber uma notificação de abandono. Isto é, quando o adversário remoto abandona a partida, o usuário local recebe uma notificação por parte do DOG server. Em casos desse tipo, a partida deve ser considerada encerrada.

3.2 Requisitos não funcionais:

Requisito não funcional 1 - Tecnologia de interface gráfica para usuário: O programa terá sua interface gráfica baseada em TKinter;

Requisito não funcional 2 - Suporte para especificação de projeto UML: O projeto deve ser especificado com o uso da ferramenta Visual Paradigm;

Requisito não funcional 3 - Interface do programa: A interface do programa seguirá o esboço abaixo:



Apêndice: Regras do jogo Ataxx

O jogo Ataxx é disputado entre 2 jogadores em um tabuleiro de 49 posições, organizado como uma matriz 7x7, e o objetivo dos jogadores é terminar a partida com o maior número de peças no tabuleiro. Ao longo da partida, os jogadores tentam expandir sua presença no tabuleiro e converter peças do adversário para sua própria cor, por meio de movimentos estratégicos.

Elementos do jogo:

Além do tabuleiro 7x7, o jogo conta com peças de cor distinta para cada jogador, representadas por discos amarelos e vermelhos. Cada jogador inicia a partida com duas peças, colocadas nas diagonais opostas do tabuleiro. Durante a partida, os jogadores irão realizar movimentos que podem duplicar suas peças ou reposicioná-las para alcançar novas áreas do tabuleiro.

Lances dos jogadores:

Os jogadores se alternam nos turnos (sendo o jogador com as peças amarelas quem inicia a partida) e, em sua vez, o jogador pode realizar um dos seguintes tipos de movimento: duplicar uma peça ou mover uma peça.

Duplicar uma peça: consiste em selecionar uma de suas peças já presente no tabuleiro e criar uma cópia dela em uma casa adjacente — ou seja, a uma distância de 1 unidade, seja na horizontal, vertical ou diagonal. A peça original permanece no local de origem, e a nova peça passa a ocupar a casa adjacente. Esse é o movimento mais comum e geralmente mais seguro, pois aumenta o número de peças no tabuleiro.

Mover uma peça: consiste em selecionar uma de suas peças no tabuleiro e movê-la para uma casa a duas posições de distância, em qualquer direção (horizontal, vertical, diagonal, ou em forma de L). Diferentemente do movimento de duplicação, ao mover uma

peça, a posição original é deixada vazia. Esse movimento é útil para alcançar regiões distantes do tabuleiro, mas não aumenta a quantidade total de peças do jogador.

Se o jogador não tiver nenhum movimento legal possível, ele deve passar a vez. Essa é a única situação em que o jogador pode deixar de fazer uma jogada durante seu turno.

Após qualquer tipo de movimento, todas as peças adjacentes (nas 8 direções) à casa de destino que forem da cor do adversário são convertidas para a cor do jogador que acabou de se mover. Isso cria uma dinâmica de constante disputa por território e influência sobre as casas do tabuleiro.

Encerramento de partida:

A partida encerra quando o tabuleiro estiver completamente preenchido, ou quando um dos jogadores não tiver mais peças. Ao final, o número de peças de cada jogador no tabuleiro é contado, e vence aquele que possuir mais peças. A mecânica de conversão de peças adversárias torna o jogo dinâmico e imprevisível, exigindo dos jogadores raciocínio tático e planejamento de médio prazo. Estratégias defensivas e ofensivas devem ser bem equilibradas para maximizar o controle do tabuleiro e evitar jogadas que resultem em grandes conversões para o oponente.