Laboratório de Redes - Trabalho 1

Arthur Aguiar Menezes de Souza

1. Introdução

Este relatório possui como propósito detalhar o desenvolvimento e o funcionamento de uma aplicação cliente-servidor para jogos de *Tic Tac Toe* ("jogo da velha"). A seção 2 apresenta detalhes do desenvolvimento do projeto, relacionado-o com os aspectos de redes de computadores e concorrência pertinentes ao escopo. Por fim, a seção 3 demonstra o funcionamento da aplicação, desde o início de um partida até a sua conclusão.

2. Desenvolvimento

Foi utilizada a linguagem de programação Java. De forma complementar, a interface de programação RMI foi usada a fim de garantir que a invocação de chamadas remotas possam ser executadas.

Os arquivos que compõem o projeto Java são os seguintes:

Arquivo	Tipo	Descrição	Relacionamento
Server	Classe	Aplicação servidor para os clientes se conectarem.	- TicTacToe
Client	Classe	Os clientes que irão se conectar ao servidor em questão serão instâncias desta classe.	- ITicTacToe
TicTacToe	Classe	Implementação do jogo	- ITicTacToe - Match - Player
ITicTacToe	Interface	Interface a ser implementada	
Match	Classe	Representa uma partida com todos os seus atributos e operações possíveis	- Player
Player	Classe	Representa um jogador com todos os seus atributos e operações possíveis.	- Match

Através da utilização do Java RMI, a aplicação passa a utilizar uma conexão TCP entre o servidor e seus clientes. Por motivos de segurança de rede, foram realizadas algumas parametrizações a fim de que estabelecer alguns limites. Contra a possível exploração de ataques DDoS, por exemplo, foi definida uma quantidade limite de partidas (e, consequentemente, jogadores) permitidas no servidor. Outra medida tomada foi a elaboração de um *garbage collector* responsável por eliminar partidas encerradas e

jogadores não mais presentes. Esse *garbage collector* elimina os objetos do servidor, removendo-os das listas de partidas e jogadores, respectivamente.

Diversas verificações de controle foram implementadas a fim de garantir que o usuário não tentará, por exemplo, acessar uma posição inexistente ou tentar posicionar sua peça em uma posição já ocupada. Como o tabuleiro foi implementado como uma matriz, essas tarefas foram facilmente desenvolvidas. Abaixo, pode-se ver algumas das verificações realizadas pela aplicação no momento em que um cliente tentar posicionar um peça:

```
if ((linha < 0) || (linha > 2) || (coluna < 0) || (coluna > 2)) {
    return 0;
}

if (board[linha][coluna] != VAZIO) {
    return 0;
}
```

A fim de garantir o correto funcionamento da aplicação em relação à sincronia, realizou-se a criação de alguns pares de *locks.* Eles são os responsáveis por proporcionarem suporte à múltiplas *threads* para a aplicação. Suas definições podem ser visualizadas abaixo:

```
ReadWriteLock uid_readWriteLock = new ReentrantReadWriteLock();
Lock id_writeLock = uid_readWriteLock.writeLock();

ReadWriteLock players_readWriteLock = new ReentrantReadWriteLock();
Lock players_readLock = players_readWriteLock.readLock();
Lock players_writeLock = players_readWriteLock.writeLock();

ReadWriteLock matches_readWriteLock = new ReentrantReadWriteLock();
Lock matches_readLock = matches_readWriteLock.readLock();
Lock matches_writeLock = matches_readWriteLock.writeLock();
```

No caso de *ReadWriteLock* “uid_readWriteLock” reservado aos IDs de jogadores, criou-se apenas um *lock* para escrita, já que não seria necessário lidar com concorrência em um cenário de leitura de IDs.

3. Funcionamento

Após a inicialização do servidor, sem nenhuma passagem de parâmetros ao *main*, pode-se executar clientes da seguinte forma:

```
java Client <ip-do-servidor> <nome-do-jogador>
```

Depois que um cliente for executado, uma partida será criada, ele será alocado a essa partida, e ela então será adicionada à lista de partidas do servidor. Após a inicialização do servidor, seu terminal deverá apresentar as seguintes mensagens:

```
RMI Registry ready!
TicTacToe server ready!
Garbage Collector!
```

Enquanto isso, após a execução do primeiro cliente, ele (o cliente) deverá visualizar algo semelhante ao demonstrado abaixo:

```
java Client 127.0.0.1 Arthur
Procurando partida …
```

Se demorar muito tempo até que outro jogador acesse o servidor, uma mensagem de tempo esgotado será exibida ao cliente. Porém, caso outro jogador acesse o servidor, o primeiro jogador passará a visualizar o tabuleiro de *tic tac toe:*

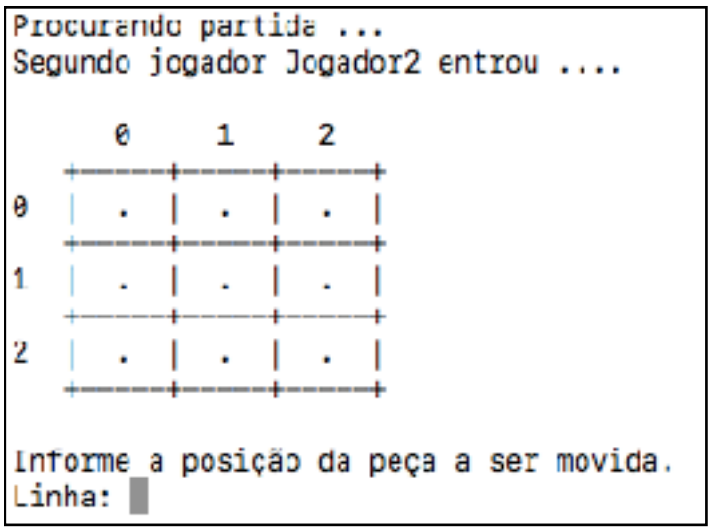


Figura 1: Interface com o usuário

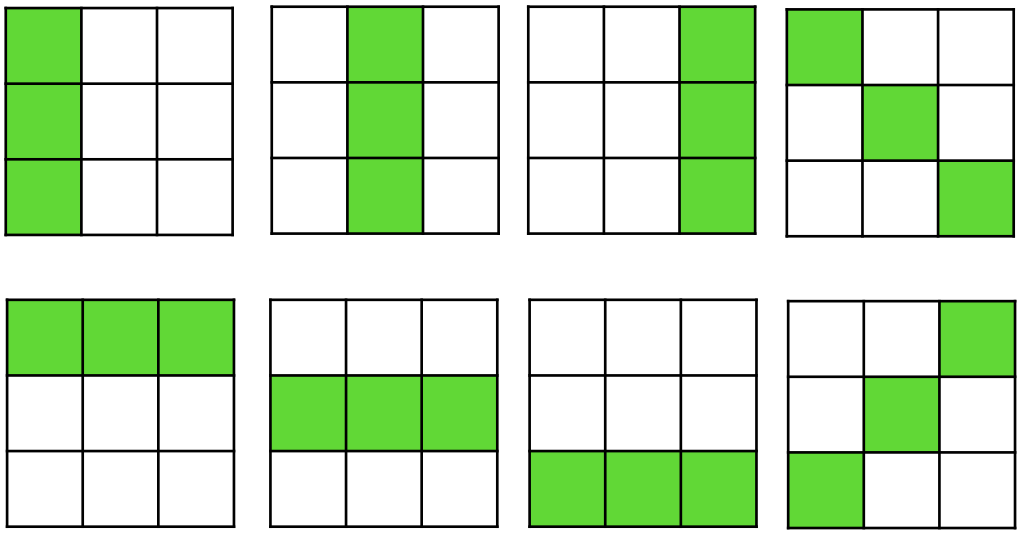
Após essa introdução, o jogador deverá, conforme demonstrado acima, informar a linha na qual deseja inserir sua peça. Em seguida, a coluna. Depois disso, o tabuleiro atualizado é demonstrado ao jogador, que deverá aguardar enquanto o segundo jogador (que também recebe o tabuleiro atualizado) realiza o seu movimento. Este mecanismo se propaga ao decorrer da partida. Sempre um jogador esperando o outro informar linha e coluna da nova peça. Lembrando que, o primeiro jogador a entrar na partida sempre comandará as peças X, enquanto o segundo jogador sempre comandará as peças O.

Quando um jogador vence a partida, ele é parabenizado, enquanto o outro é informado de sua derrota, como demonstram as figuras abaixo:

Figura 2: Vencedor

Figura 3: Perdedor

As condições de vitória são as seguintes:



Para jogar uma nova partida, basta executar o cliente novamente que, então, ele será alocado a uma nova partida.