Universidade Federal de Viçosa – Campus Florestal

ROTEIRO TRABALHO PRÁTICO (EM DUPLA)
BATALHA NAVAL

Curso: Bacharelado em Ciência da Computação
Disciplina: Redes de Computadores
Professora: Thais R. M. Braga
Data de Entrega (via PVANet): 03/07/2019 **– Data das Entrevistas:** 05/07/2019
Valor: 20 pontos

Neste trabalho, definiremos um pequeno protocolo da camada de aplicação que padroniza as regras de um jogo, e o implementaremos através dos soquetes de Berkeley. Em relação a pilha de protocolos, utilizaremos IP na camada de rede e TCP na camada de transporte. O jogo é simples e permitirá apenas 2 jogadores, sendo que os mesmos só podem realizar ações uma vez liberados pelo Servidor. O Servidor é o elemento do jogo que controla toda a lógica e organiza as ações dos jogadores.

No jogo a ser implementado, uma partida é vista como o enfrentamento entre dois jogadores, até que um deles ganhe ou que seja declarado empate. Durante uma partida, cada ação de um dos jogadores será chamada de jogada.

- **Batalha Naval**

A aplicação a ser implementada neste trabalho é do jogo conhecido como Batalha Naval. Este é um jogo bastante conhecido e muito simples, no qual dois adversários possuem um mapa onde estão dispostas embarcações em determinadas posições (coordenadas) e eles tentam afundar os barcos um do outro lançando mísseis em coordenadas específicas. Uma embarcação pode ocupar uma ou mais coordenadas, dependendo do seu tamanho. Ganha o jogador que conseguir afundar o maior número de embarcações adversárias.

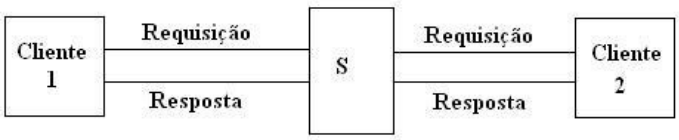
Em cada jogada, um dos jogadores escolhe uma coordenada válida (dentro dos limites do mapa e que ainda não foi utilizada) no mapa de seu adversário. Se a coordenada coincide com uma parte de uma das embarcações do adversário, um acerto é contabilizado. Caso contrário, a tentativa é considerada um erro. Um barco afunda no jogo quando o adversário consegue acertar todas as suas partes. Vamos considerar que o mapa de cada jogador é uma matriz com dimensões NxN (cada coordenada no mapa é portanto uma posição da matriz), que apenas barcos de um tamanho X fixo possam ser criados no jogo e que cada jogador possuirá exatamente Y embarcações, aleatoriamente dispostas na matriz. Também vamos considerar que cada jogador possui um número Z de tentativas para afundar todas as embarcações do adversário (note que, tanto NxN como Z não podem ser menores do que Y*N). Cada uma dessas variáveis deverá ser configurável no jogo. A partida termina quando um dos jogadores afunda todas as embarcações de seu adversário antes das Z jogadas disponíveis ou quando os mesmos esgotam o número de tentativas disponibilizado. No primeiro caso, o jogador vitorioso é aquele que destruiu as embarcações de seu oponente. No segundo caso, será aquele com maior número de acertos. Veja que é possível, no segundo caso, a ocorrência de um empate.

A implementação deste jogo seguirá o modelo Cliente/Servidor. A entidade que controla as regras do jogo, ou seja, o protocolo a ser seguido, é o Servidor. Os jogadores

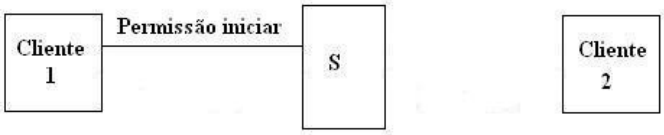
serão os Clientes. Cada partida somente poderá conter dois jogadores ou clientes. Assim, dois clientes devem enviar um pedido de participação em uma partida ao servidor para que a mesma seja iniciada. Cada jogada solicitada pelos clientes é enviada ao servidor. A seguir estão descritas em forma de regras os modelos para inicialização, condução e término da partida. As regras de condução definem quais são as jogadas válidas:

1 – Primeiramente o Servidor deve ser inicializado. Neste caso, ele estará executando em uma máquina com endereço IP e porta conhecidos pelos clientes. O servidor espera pelos pedidos de conexão dos clientes. O servidor controlará a partida. Somente dois jogadores (clientes) serão aceitos por partida. Para fins de implementação, pode ser considerado que nunca mais do que dois clientes farão a solicitação de conexão.

2 – Um cliente, quando inicializado, envia pedido de conexão ao servidor. O servidor cria o mapa (matriz) do jogador, enviando-a ao mesmo em uma mensagem de resposta. O cliente constrói sua interface inicial e aguarda a próxima mensagem do servidor, que vai liberá-lo para enviar sua jogada. A interface do jogo para os clientes deverá possuir uma tela bipartida, cada lado representando o mapa de um dos jogadores. De um lado o mapa do próprio jogador, visualizando todas as suas embarcações, com as posições já atingidas claramente demarcadas. Do outro lado, o mapa do oponente mostra apenas as posições que foram usadas pelo jogador em suas jogadas, com a respectiva resposta recebida do servidor. Nesta interface devem constar o número de jogadas restantes que o jogador possui, bem como o número de acertos e o de embarcações oponentes já afundadas. Quando dois clientes executarem o passo anterior (2), forem aceitos pelo servidor e uma partida for criada para eles, o jogo poder começar. Neste momento ambos os jogadores estão confirmados.



3- O servidor manda uma mensagem para o cliente inicial, que será sempre o jogador 1, permitindo o início do jogo.



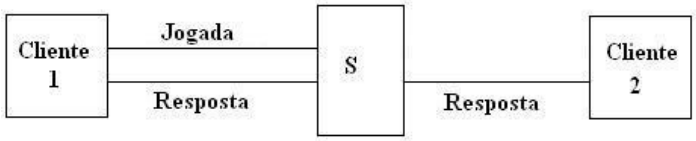
3 – O cliente escolhido como inicial pode enviar sua jogada ao servidor, a partir do momento em que recebeu permissão. Uma jogada deve conter a coordenada escolhida pelo jogador para o envio do míssel no mapa de seu oponente. Enquanto um jogador estiver habilitado para jogar, sua interface estará liberada. Entretanto, o outro jogador, ainda não liberado, deverá ter sua interface travada/congelada.

> 3.1 - A jogada recebida pelo servidor é registrada por ele, isto é, ele registra no mapa do jogador e de seu oponente o resultado do míssel enviado, atualiza o placar e, ao final, uma mensagem é enviada a ambos os jogadores. Essa mensagem contém a aceitação da jogada, bem como a atualização dos mapas e

do placar. A interface de ambos os jogadores deve ser atualizada. Caso com esta jogada o jogador acerte a última parte de uma embarcação, no placar, essa informação deve ser contabilizada. Caso com esta jogada, a última embarcação do oponente tenha sido afundada, antes do término do número mínimo de jogadas, o Servidor envia também na mensagem o anúncio do vencedor e de término do jogo. Caso com esta jogada, o jogador não tenha afundado todas as embarcações do adversário, e ela seja a última jogada possível do segundo jogador, o servidor envia anúncio de empate e término do jogo. O jogador atual, após receber esta mensagem de atualização do servidor, caso o jogo não tenha terminado, congela sua interface.

3.2 – Se o jogo não tiver terminado, após a mensagem anterior, o Servidor envia mensagem para o jogador número 2, liberando-o para enviar sua jogada.

4 – O segundo cliente, que estava parado aguardando a vez de seu adversário, ao receber a mensagem de liberação, fica autorizado a calcular sua jogada e enviá-la ao servidor. O servidor então procede exatamente como fez para o primeiro jogador, enviando novamente as mensagens para atualização da interface gráfica. Após essa mensagem, caso não tenha ocorrido término do jogo, o jogador número 2 congela sua interface. O processo volta ao item 3 e se repete, até o término do jogo.



• Implementação

Para implementar esse trabalho, você deve criar um programa cliente e outro servidor, utilizando soquetes de Berkeley em uma linguagem escolhida a critério da dupla. A ferramenta de desenvolvimento poderá ser aquela de sua preferência. Todo o desenvolvimento deve ser feito em sistema operacional Linux.

Não é deverão ser utilizadas threads para o desenvolvimento deste trabalho. As ações são todas sequenciais entre clientes e servidor e, desta forma, nenhum deles terá nada a ser processado de maneira paralela/pseudo-paralela. Definitivamente não há qualquer razão para se usar serialização de objetos. Cada mensagem deve conter apenas os dados necessários, em formato numérico ou textual.

• Entrega

Cada dupla deve entregar, através do PVANet, um arquivo compactado contendo código fonte e pequena documentação do seu trabalho prático (2 ou 3 páginas), contemplando as seguintes informações:

- Nome dos componentes da dupla e a turma
- Breve explicação sobre o objetivo do trabalho
- Como foi desenvolvido o programa cliente, com a descrição dos principais métodos e nome do componente responsável por ele.
- Como foi desenvolvido o programa servidor, com a descrição dos principais métodos e nome do componente responsável por ele.
- Exemplos de utilização