Uma experiencia com jogos distribuídos em rede

Renato Domingues, Diego Castro
Informatics and Comp. Science Dept.
Rio de Janeiro State University
Rio de Janeiro, Brazil
(renato95, diegocbcastro)@gmail.com

Alexandre Solon Nery
Department of Electrical Engineering
University of Brasilia
Rio de Janeiro, Brazil
anery@redes.unb.br

***Abstract*—Com o passar dos anos os jogos estão ficando cada vez mais complexos, podendo agora comunicar pessoas que estejam remotamente distantes, criando assim uma nova ramificação para os jogos, chamados de jogos distribuídos ou multiplayers. O artigo aprensenta uma breve comparação entre alguns meios de construção e problemas relacionados a construção de jogos distruídos e faz utilização de um desses meios para criação de um simples jogo.**

***Keywords*-Jogos; Jogos distribuidos; Protocolos de comunicação; Protocolos de rede;**

I. Introdução

Jogos são atividades voluntárias, que se utilizam de um mundo abstrato onde um ou mais jogadores interagem com o ambiente para alcançar um objetivo, estejam os jogadores em conflito ou não [1]. Uma das ramificações de jogos são os jogos distribuidos ou multiplayer que através do envio de mensagens pela rede possibilitam que jogadores remotamente distantes interajam em um mesmo ambiente [2]. Jogos desse tipo utilizam a rede como meio de comunicação, entretanto, podem sofrer com diversos problemas, como: simultâneadade das informações entre os jogadores, sobrecarga de rede, latência, largura de banda e perda de pacotes [2] [3].

No início da internet a comunicação entre os computadores através da rede ocorria por meio da troca de mensagens, os jogos de computadores que já existiam na época também começaram a se comunicar entre sí através da utilização de protocolos de rede, sendo hoje os mais utilizados para jogos o UDP e o TCP. [2].

O artigo apresenta uma breve comparação entre as possíveis maneiras de construção de um jogo distribuído, onde uma dessas abordagens foi experimentada para construção de um simples jogo inspirado no antigo Arcade Pac-Man. O artigo foi dividido da seguinte maneira: a seção II apresenta uma breve contextualização sobre os meios de construção de jogo distribuido, a seção III apresenta o estudo de caso da criação de um simples jogo, a seção IV demonstra algumas limitações encontradas e os trabalho futuros do trabalho e por fim a seção V apresenta algumas considerações finais.

II. Contextualização

A comunicação através das redes costuma ser dividida em camadas, sendo cada uma responsável por um aspecto diferente da comunicação. Segundo o modelo OSI (Open System Interconnection) que é o padrão de rede de referência da ISO, existem 7 camadas [4] [5]:

- Física: Corresponde a parte física da rede, como: cabos, fibras opticas, etc [4] [5].
- Enlace: Sua função principal é a codificação e decodificação dos dados que são transmitidos pela rede [4] [5].
- Rede: Transmite os pacotes de um nó na rede para outro, utilizando o protocolo IP (Internet Protocol) que é o responsável pela identificação de nó na rede [4] [5].
- Transporte: Trabalho no fluxo de dados da transmissão das informações.
- Sessão: Gerencia as conexões entre as aplicações [4] [5].
- Apresentação: Normaliza os dados que são enviados de um nó para outro através da camada de transporte [4] [5].
- Aplicação: Acontece no no topo da camada OSI processando os dados recebidos e demonstrando para o usuário [4] [5].

Como já foi dito na introdução do artigo, os jogos que se comunicam através da rede podem enfrentar alguns problemas, que serão descritos a seguir.

A. Dificuldades

- Baixa largura de banda: baixa quantidade de dados que trafegam pela rede em um determinado período de tempo [2].
- Alta latência: É quantidade de tempo que uma mensagem leva para sair da origem e chegar ao destino, o que impacta no tempo de visualização de uma determinada ação pelo jogador adversário [2].
- Alto jitter: alta variação da latência, o que impacta na dificuldade de balanceamento da latência [6].
- Perda de pacote: as informações que saíram da origem não chegaram corretamente ao destino, havendo perda de informações [2].

As dificuldades citadas acima podem influenciar diretamente em um jogo distruibuído, por diversos fatores, como: problemas na infra-estrtura, congestionamento de dados, limite da velocidade da rede, etc. Essas dificuldades devem ser controladores de maneira cuidadosa pelo criador do jogo para que o usuário possa ter uma boa experiencia, caso contrário, o jogo pode executar sem qualidade, com baixa perda de frames, etc. Existem alguns protocolos e modelo de comunicaçãoque tentam contor nar essas variáveis, tentando dar uma melhor de troca de informações através da rede.

B. Protocolos

Os protocolos de comunicação atuam na camada de transporte, sendo responsáveis por definir como ocorre a troca de mensagem entre dois nós. Entretanto, quem faz o papel de identificar o caminho que será percorrido de um pacote do ponto A até o B é a camada de rede. Para que essa comunicação aconteça é necessário que cada um dos nós da rede utilize um protocolo específico e tenha um socket acessível que é uma combinação do endereço IP do computador associado a uma porta [4]. Os protocolos mais utilizados para transporte de mensagens via rede são os TCP (Transmission Control Protocol) e o UDP (User Datagram Protocol) com outro protocolos sendo criados por meio deles. O primeiro deles é o modelo utilizado na Internet que possui quatro camadas, onde as camadas de apresentação e de sessão desapareceram do modelo [2] [7]. Este protocolo possui algumas características, como: ordenação e não duplicação dos pacote com entrega garantida que é possível pelo fato de que o receptor enviar uma confirmação chamada de ACK para o remetente, sendo que este só envia o próxima pacote se receber a confimação, entretanto apesar dessas vantagens ele possui um cabeçalho maior em relação ao UDP, possuindo uma performance menor. O segundo, tem como características: ser rápido, não possuir garantia contra perdas e é usado para ocasiões aonde a perda de pacotes é tolerável [2] [7]. A seguir uma a Fig. 1 e a Tab. 1 mostram um comparativo entre as duas abordagens.

Como já foi dito alguns protocolos podem ser construídos através desses dois protocolos mensionados anteriormente como, por exemplo, o RPC (Remote Procedure Call), RMI (Remote Method Invocation). O RPC é um protocolo para invocação de um procedimento em um maquina remota [8]. Já o RMI é uma abstração da plataforma Java para executar funcionalidades para objetos distribuídos por meio de chamadas RPC [9].

Cada uma das aplicaçãoes que implementam um dos protocolos citados acima devem utilizar algum modelo de comunicação para conectar os usuários na rede.

C. Modelo de Comunicação

- Peer-to-Peer(P2P): Topologia onde cada usuário estabelece uma conexão com todos os outros usuários. Características: bom tempo de resposta, comunicação redundante, pouco escalável e não possui um controle central. [7]
- Cliente-Servidor: Existe um servidor central que controla todas as informações e as envia para os usuários. Características: razoavelmente escalável, possui um controlador central, topologia estática, trabalha com posicionamento geográfico, quanto mais longe o cliente está do servido, maior o tempo de resposta. [7]
- Híbrida: Mistura as características das duas citadas acima. Características: escalável, uniforme em resposta temporal, possui um controlador central, comunicação redundante e possui uma topologia estática. [7]

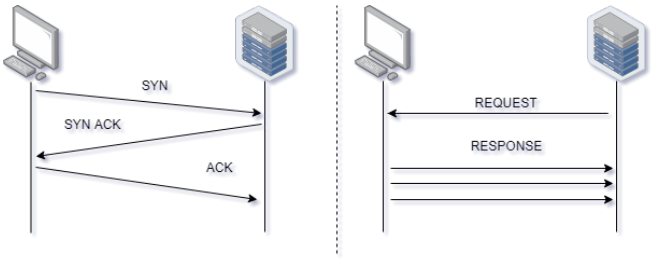

Figura 1. Protocolo TCP e UDP [10].

Tabela I. Tabela comparativa de protocolos [11] [7]

	TCP	UDP
Conexão	É baseado em conexão.	Não existe conceito de conexão.
Confiabilidade	Garante a entrega de todos os pacotes enviados, na ordem em que foram enviados. Controle de pacotes por meio de reenvio.	Não garante a entrega do pacote, nem a ordem.
Pacotes	Gera os pacotes automaticamente conforme a necessidade.	A criação dos pacotes é implementada na aplicação
Escalabilidade	Controla o fluxo de dados automaticamente.	O controle do fluxo é feita pela aplicação.
Performance	Por da garantia do envio dos pacotes possui uma performance menor.	Alta

III. Estudo de caso

Para verificar o potêncial dos jogos distribuídos uma experiencia de desenvolvimento foi realizada buscando produzir um jogo multi-player sendo o mais rápido e prático possível em relação a tempo e complexidade de desenvolvimento. Devido as informações citadas nas seções anteriores optou-se pela criação do jogo utilizando o protocolo de rede RMI (devido sua facilidade de uso) integrado com o modelo de comunicação cliente-servidor por possuir boas características e utilizar um controlador central onde seria mais fácil de controlar a troca de mensagens. Por causa das características escolhidas para a criação do jogo, escolheu-

se como tema da aplicação o jogo Pac-Man onde o servidor seria o Pan-Man e cada um dos cliente seria um fantasma.

Para esse desenvolvimento utilizou a ideia de construir um servidor com o menor número de responsabilidades possíveis, tendo este a tarefa somente de fazer gerenciamentos dos objetos do jogo, deixando a regras do jogo sendo controladas por cada um dos clientes.

O jogo inicia quando o cliente através de uma combobox escolhe um fantasmas entre as opções possíveis para jogar, a partir daí o jogo fica em loop esperando que os outros jogadores se posicionem. O jogo funciona a partir de uma lista de 5 posições, que é formada por 4 jogadores e o controlador central, a cada vez que um jogador entra na partida uma posição dessa lista é preenchida e enviada para o servidor que reenvia para todos os outros cliente que assim ficam sabendo que mais um jogador entrou no partida. Para o controle dessa lista 3 funções foram implementas:

- SentEnt: envia as informações como posição e velocidade do cliente para o servidor;
- RecvEnt: envia as informações dos outros usuários para um cliente para este saber onde seus adversários estão;
- Ready: verifica se os cinco jogadores estão conectados e jogo pode ser iniciado.

A conexão do jogo foi feita para tentar ser a mias leve possível dentro das características que estavam sendo utilizadas, por este motivo, o objeto do cliente permance vivo (sem conexão com o servidor) no jogo por no máximo 5 segundos, caso contrário o mesmo retorna nulo para não atrapalhar na experiência de jogos dos outros usuários. Cada cliente informa apenas seu próprio objeto para o servidor e lê o de seus adversários.

Cada um dos personagens do jogo é uma instância de uma mesma classe, o que diferencia o Pac-Man dos outros jogador é sua posição na lista inicia, sendo semrpe a primiera posição da lista. O mapa do jogo é formado por uma matriz 2x2, onde cada um dos números representa um bloco de 30x30 pixels, vale lembrar que cada um dos números tem seu significado específico, sendo ele um personagem, uma parede ou uma moeda do jogo. Na Fig. 2 é possível observar o mapa do jogo.

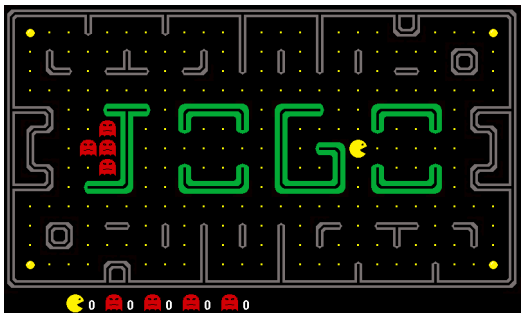

Figura 2. Mapa do jogo.

As colisões do jogo são verificadas apenas quando o Pac-Man entra em contato com alguns dos fantasmas, ou quando um personagem entra em contato com algumas parede, ou seja, quando ele está a menos de 5 pixels de distância. As colisões são testadas a cada 30 pixels (que é o tamanho do sprite) em todas as 4 posições.

Por fim, a potuação do jogo funciona da seguinte maneira: o usuário que jogar com Pac-Man completa 1 ponto assim que conseguir coletar todas as moedas do jogo sem ser capturado por nenhum fantasma, já o fantasma ganha 1 ponto toda vez que capturar o Pac-Man. Para deixar o jogo um pouco mais justo, o Pac-Man foi implementado com uma velocidade 1.5 vezes a do fantasma.

IV. Trabalhos Futuros e Limitações

Como o jogo foi construído para ser o mais simples possível algumas limitações foram encontradas e seram descritas a seguir:

- Um dos motivos para escolha do RMI como protocolo de comnunicação foi a facilidade de implementação por parte do desenvolvedor;
- O jogo foi construído como pré-requisito para aprovação em uma matéria, portanto o tempo de desenvolvimento ficou limitado ao tempo da disciplina;
- Alguns problemas de sincronização também foram identificados.

Alguns trabalhos futuros são esperados para melhoria do jogo atual como: a possibilidade de utilização de inteligência artificial para preencher os jogadores que não estiverem na partida, melhoria da sincronização do jogo e experimentação de outro protocolos.

V. Considerações Finais

O artigo apresentou uma experimentação sobre a construção de jogos distribuídos fazendo breve comparação entre os métodos mais utilizados para construção desses jogos e em seguida escolheu um desses meios para construção de um jogo simples para por em prática o que foi encontrado. A partir do que foi pesquisa e construído foi possível observar que os jogos distribuídos possuem algumas características e dificuldade particulares que devem ser tratadas cuidadosamente para o jogador possuir uma experiência.

References

[1] G. Xexéo, A. Carmo, A. Acioli, B. Taucei, C. DIpolitto, E. Mangeli, J. Kritz, L. Costa, and R. Monclar, “O que são jogos?” 2017.

[2] I. L. Picoli, “Arquitetura cliente-servidor em jogos multiplayer,” B.S. thesis, Universidade Tecnológica Federal do Paraná, 2011.

[3] R. Villa and A. S. Felinto, “Arquiteturas de rede para jogos mmo e técnicas para amenizar os problemas de rede.”

[4] R. Scaroni, "Desenvolvimento de um protocolo de rede local p2p e de um plugin para o motor de jogos unity 3d," Ph.D. dissertation, IME-USP, 2014.

[5] H. Zimmermann, "Osi reference model-the iso model of architecture for open systems interconnection," *IEEE Transactions on communications*, vol. 28, no. 4, pp. 425–432, 1980.

[6] D. C. Verma, H. Zhang, and D. Ferrari, "Delay jitter control for real-time communication in a packet switching network," in *Proceedings of TRICOMM91: IEEE Conference on Communications Software: Communications for Distributed Applications and Systems*. IEEE, 1991, pp. 35–43.

[7] J. M. I. Queirós *et al.*, "Sistema de caracterização da rede em jogos 3d online multi-jogador," 2009.

[8] R. Srinivasan, "Rpc: Remote procedure call protocol specification version 2," Tech. Rep., 1995.

[9] E. Pitt and K. McNiff, *Java. rmi: The Remote Method Invocation Guide*. Addison-Wesley Longman Publishing Co., Inc., 2001.

[10] "Tcp/ip vs udp: What's the difference?" https://www.colocationamerica.com/blog/tcp-ip-vs-udp, accessed: 2019-05-19.

[11] "Desenvolvimento de jogos em rede: Protocolo udp," https://www.fabricadejogos.net/posts/desenvolvimento-de-jogos-em-rede-protocolo-udp/, accessed: 2019-06-01.