Whiteboard Peer-to-Peer

Bruno Mendes de Souza, Mairieli Santos Wessel

Resumo—Sistemas peer-to-peer (P2P) suportam aplicativos e serviços distribuídos e os dados e recursos computacionais nesses sistemas são resultantes da colaboração de máquinas na Internet. Desta forma será apresentada a construção, desde a arquitetura até a implementação, de um sistema whiteboard (quadro branco) colaborativo P2P.

1 Introdução

Em sistemas peer-to-peer dados e recursos computacionais são provenientes da colaboração de muitas máquinas na Internet. Desta forma, os sistemas peer-to-peer representam um paradigma para a construção de sistemas distribuídos e têm o objetivo de permitir o compartilhamento de dados e recursos em grande escala, sem a necessidade de ser administrado de forma centralizada.

O projeto proposto e implementado neste trabalho, construído a partir de um sistema peer-to-peer, é um whiteboard (quadro branco) colaborativo. O whiteboard é um quadro branco digital, que segue o conceito de um quadro branco físico permitindo escrever e desenhar de forma colaborativa.

O trabalho está organizado da seguinte forma, na seção 2 será apresentado o referencial teórico sobre sistemas peer-to-peer. Na seção 3, o método utilizado para a implementação do whiteboard e as bibliotecas utilizadas serão descritas. Os resultados da implementação estão na seção 4, e por fim a conclusão é apresentada na seção 5.

2 Referencial Teórico

Os sistemas peer-to-peer têm como objetivo suportar serviços e aplicativos distribuídos. Os processos (peers) interagem de forma cooperativa para realizar as atividades distribuídas, desempenhando funções simétricas, desta forma não há distinção entre cliente e servidor.

Os sistemas e aplicativos peer-to-peer passaram por três gerações de desenvolvimento. A primeira geração foi lançada pelo Napster, um serviço de troca de músicas, e a principal característica eram índices centralizados e replicados. A segunda geração é caracterizada por oferecer maior escalabilidade, anonimato e tolerância a falhas. Na terceira geração apareceram camadas de middleware para o gerenciamento dos recursos distribuídos.

Algoritmos para a distribuição e localização de objetos são um aspecto importante do projeto de um sistema peer-to-peer. O objetivo é distribuir um serviço descentralizado e organizado, equilibrando as cargas de armazenamento e processamento entre os computadores, e garantir disponibilidade à medida que máquinas entram e saem do sistema.

3 Método

O whiteboard foi desenvolvido em Python versão 3.5.2, utilizando mecanismos como threads e soquetes. A interface gráfica do whiteboard foi implementada através da biblioteca pygame. Pygame é um módulo python que fornece a API da biblioteca Simple DirectMedia (SDL), uma biblioteca de multimídia, e outras funcionalidades que possibilitam a programação gráfica. Para implementar a interface de conexão inicial foi utilizado o módulo pythondialog que oferece uma forma simples de criar aplicações gráficas em modo texto.

3.1 Modelo do Sistema P2P

O modelo escolhido para a implementação do Whiteboard P2P consiste em um Servidor e diferentes grupos de quadros cooperativos.

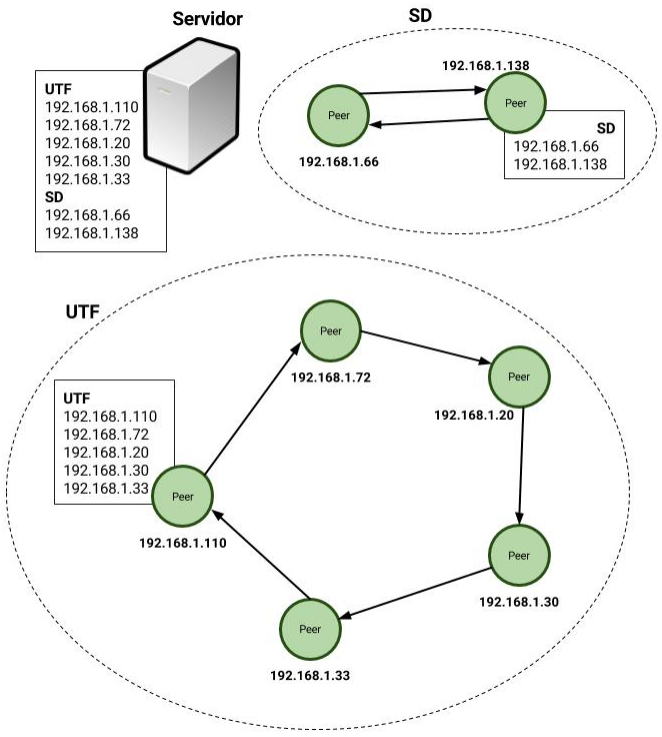

Figura 1: Modelo do Sistema P2P.

O Servidor mantém uma lista com os quadros disponíveis e o IP dos peers conectados a esses quadros. O servidor mantém a lista de peers o mais atualizada possível recebendo mensagens dos peers que indicam se algum peer caiu e não está mais disponível.

Os grupos de quadros cooperativos são organizados em formato de anel. Todos os peers mantém a lista dos peers

conectados no grupo atual. Cada peer envia dados para o próximo da lista e o último envia dados para o primeiro. Por exemplo, na Figura 1, o peer 192.168.1.33 realiza um desenho no seu quadro e envia para o 192.168.1.110, este propaga o desenho para o 192.168.1.72 e assim consecutivamente até chegar novamente no 192.168.1.33, o qual para de propagar seu próprio desenho.

Todo peer estabelece inicialmente uma cor aleatória RGB para seus desenhos, a identificação do desenho de cada peer na rede é identificada por esta cor. Assim, se o peer recebe um desenho com sua própria cor, sabe que não é mais necessário propagá-lo.

Para um peer entrar em grupo ele realiza uma consulta ao servidor, podendo criar um novo grupo definido por um nome ou entrar em um grupo já existente. No caso de se conectar a um grupo já existente, o servidor o envia a lista de peers e este vira o último da lista.

No caso de um peer não estar mais disponível, o peer que estava mandando mensagens a ele se conecta ao próximo da lista e informando o IP que não está mais disponível, notificando também ao servidor.

3.2 Arquitetura do Peer

Na arquitetura do Peer foram implementados seis módulos, que são o Peer, Dial, Listener_UDP, Listener, Sender e Whiteboard. O módulo Peer é a classe inicial e possui os principais métodos para a troca de mensagem com o servidor. Inicialmente, o Peer escolhe uma cor para o representar no grupo e instância o módulo Dial.

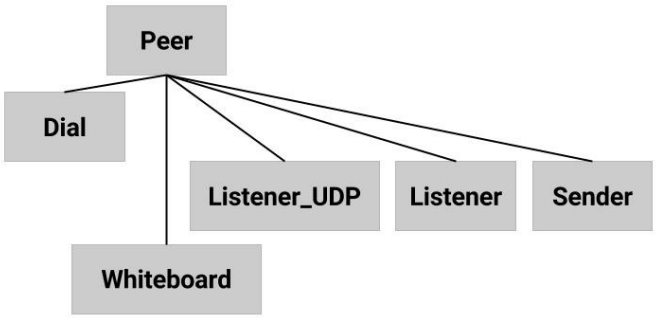

Figura 2: Arquitetura do Peer.

No módulo Dial se encontra a interface que possibilita entrar em um grupo de quadros colaborativos. É neste momento que a biblioteca pythondialog é utilizada para implementar a interface de comunicação com o usuário em modo texto. O Dial utiliza os métodos do Peer para trocar mensagem com o servidor. Os métodos utilizados são para conectar ao servidor, requisitar os grupos de quadros cooperativos disponíveis, criar um novo grupo e se conectar a um grupo.

Ao entrar em um grupo, o Peer instancia uma Thread Listener que fica aguardando uma conexão. Quando uma conexão é estabelecida, o Listener passa a receber desenhos tcp na porta 5001. Ele verifica se o desenho que acabou de receber não é próprio desenho analisando a cor, e se não for um desenho que ele mesmo desenhou, ele adiciona em duas listas queue_receiver e queue_sender. Se o peer que estava enviando desenhos cair, ele voltar aguardar uma nova conexão. Como os peers estão organizados em um anel, ele sempre vai receber os desenhos por meio do anterior dele na lista.

Após instanciar uma Thread Listener, se ele estiver entrando em um grupo já existente, o Peer envia uma mensagem para o primeiro do grupo avisando que ele acabou de entrar, e também envia uma mensagem para o último avisando que deve se conectar a ele.

Em seguida será instanciado o Whiteboard, que é uma Thread. Ele realiza duas funções, consome e mostra no quadro os desenhos da queue_receiver, e a medida que o desenho é feito ele guarda o desenho na queue_sender. Quando ocorre um evento de clique no botão de fechar janela, ele notifica o módulo Peer para que sejam fechados todos os soquetes de todos os módulos e para que o servidor seja notificado sobre a saída, a fim de atualizar a lista de peers.

Listener_UPD escuta mensagens udp na porta 5002. Ele recebe um connect indicando que outro peer está entrando ou um remove indicando que algum peer caiu, e a partir dessas mensagens ele atualiza a própria lista. Antes de enviar essas mensagens ao próximo peer ele verifica se a sua lista já estava atualizada com o conteúdo da mensagem, se sim ele não reenvia.

O Sender faz uma conexão tcp na porta 5001 do próximo peer no anel. Caso na lista de ips do peer tenha somente o próprio ip, ele fica aguardando até ter outro ip para poder enviar os desenhos. Caso a lista contenha outros ips, ele tenta se conectar ao próximo da lista, se ele for o último ele tenta se conectar com o primeiro, se ele não conseguir se conectar ele tenta o próximo mais um da lista. Quando ele tenta se conecta a algum peer mas não consegue, ele guarda esse ip e manda ao próximo que der certo a mensagem de remove. Ao conseguir estabelecer uma conexão ele envia os dados da queue_sender. Se ele estiver conectado com o primeiro ele sempre ficará verificando se existe um novo último, para poder parar de se conectar com o primeiro e passar a se conectar com o novo último. Caso a conexão com o próximo peer caia, ele avisa o servidor e voltar ao processo inicial de tentar se conectar a algum peer.

3.3 Arquitetura do Servidor

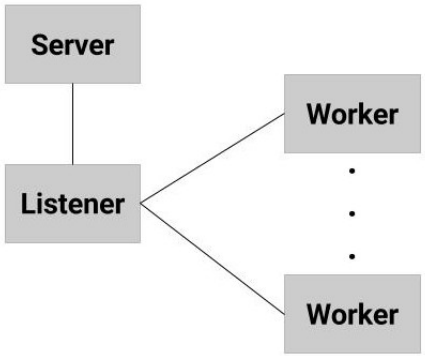

Figura 3: Arquitetura do Servidor.

Na arquitetura do Servidor foram implementados três módulos, sendo eles Server, Listener e Worker. O módulo Server é responsável por iniciar a aplicação e também iniciar

a Thread Listener. O Listener tem a responsabilidade de ficar escutando mensagens udp provenientes dos peers na porta 6000, essas mensagens serão explicadas na seção de protocolo. Ao receber uma nova requisição de um peer, o Listener, cria um novo Worker para tratar e responder essa requisição. Esse módulo Worker tem também a finalidade de realizar o log das requisições e das ações tomadas por ele.

3.4 Protocolo

As mensagens de comunicação com o servidor são especificadas da seguinte forma:

Para um peer listar os grupos disponíveis no servidor, envia-se: list all

A resposta do servidor a listagem de grupos são os nomes separados por dois pontos: <name1>:<name2>

Para se conectar a um grupo existente, o peer indica o nome do grupo: connect <name>

O servidor responde ao connect com a lista de ips do grupo separados por dois pontos: <ip1>:<ip2>:<ip3>:<ip4>

Para o peer criar um novo grupo, indica-se o nome do grupo a ser criado: create name

A requisição de criar um grupo o servidor responde com uma mensagem de confirmação: created, ou com uma mensagem indicando que o nome já está sendo utilizado: board exists

Ao se desconectar o peer envia ao servidor uma atualização com o nome do grupo em que estava: disconnect <name>

Quando um peer deseja atualizar o servidor notificando que algum outro peer não está mais disponível, é enviado a mensagem com o ip do peer desconectado e o nome grupo: remove <ip> <name>

A comunicação de atualização das listas de ips entre os peers é definida pelas seguintes mensagens:

Para notificar que um peer não está mais disponível, é enviado o seu IP: remove <ip>

Para notificar a entrada de um novo peer, é enviado o IP: connect <ip>

A transmissão de desenhos é definida por uma sequência de pontos a serem ligados e um identificador que sinaliza a quebra da sequência:

Um ponto consiste na cor em RGB e os valores de X e Y: :<r>:<g>:<b>:<x>:<y>

Uma quebra sequência de ligação de pontos é definido por uma cor RGB e o caractere X: :<r>:<g>:<b>:x

4 Resultados

Durante o processo de implementação houveram alguns problemas como o não recebimento de uma mensagem de cada vez pelo soquete tcp, em que foi necessário realizar um tratamento especial para as mensagens de desenho recebidas.

Foram realizados testes no whiteboard nas máquinas do laboratório E103, da UTFPR, após a finalização da implementação do sistema. Constatou-se que a troca de mensagem dos peers com o servidor e a troca de dados entre os peers estava correta e o sistema funcionou como o esperado.

5 Conclusão

O trabalho implementado na disciplina de Sistemas Distribuídos teve como objetivo colocar em prática os conteúdos ministrados pelo Professor Rodrigo Campiolo. Verificou-se que modelar a arquitetura é de suma importância para a correta implementação do sistema. O sistema cumpriu seu objetivo de poder disponibilizar aos clientes um quadro colaborativo, porém possui ainda alguns problemas não solucionados, como o de não replicar os desenhos já existentes no grupo para um novo peer.

Referências

[1] COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T. Sistemas Distribuídos - Conceitos e Projeto. 5.ed. Porto Alegre, RS: Bookman, 2013