P2World

Aplicação de troca de ficheiros em rede peer to peer

SISTEMAS DISTRIBUÍDOS @ ESTGF 2013/2014

8090228, LUÍS SOUSA

8090242, RICARDO BARBOSA

Rede p2p

- Contextualização

Contextualização

O sistema p2p não contém uma infraestrutura central, o que significa que não depende de um servidor para efetuar ações, em vez disso esse trabalho de servidor está distribuído por todos os *peers* presentes na rede.

Com uma arquitetura assim, o sistema fornece uma maior nível de disponibilidade e de *performance* bem como um maior nível de segurança pois não sofre ataques centralizados como seria de esperar numa arquitetura cliente-servidor.

Peer

- Funcionalidades
  - Listagem de Ficheiros
  - Troca de Ficheiros
- Conceito geral

Funcionalidades (listagem de ficheiros)

Em funções relacionadas com as listas de ficheiros, um *peer* é capaz de:

- Efetuar pedidos de listas parciais a todos os *peers* que se encontrem ligados;
- Efetuar a transferência de todas as listas parciais existentes no momento do pedido;
- Atender o pedido de um outro *peer* a requisitar a sua lista parcial;
- Enviar a sua lista parcial para um ou múltiplos *peers*;
- Construir uma lista global de ficheiros com base nas listas parciais recebidas;

Funcionalidades (troca de ficheiros)

Em funções relacionadas com trocas de ficheiros, um *peer* é capaz de:

- Efetuar um pedido a outro *peer* de um dado ficheiro presente na sua lista de ficheiros;
- Efetuar uma transferência de um ficheiro vindo de outro *peer*;
- Atender pedidos de transferência de ficheiros de um ou mais *peers*;
- Enviar ficheiros em simultâneo para múltiplos *peers*;
- Transferência de vários ficheiros em simultâneo a partir de múltiplos *peers*;

Conceito geral

Cada *peer* está conectado a uma rede que é comum a todos os *peers* presentes na nossa aplicação. A partir desta rede é possível obter a lista de ficheiros de cada *peer* bem como é possível saber e guardar informações de cada *peer* para uma futura ligação.

As informações de cada *peer* consistem em saber o seu endereço e a porta em que está a aceitar pedidos de transferência de ficheiros.

Após selecionar o ficheiro pretendido, é estabelecida uma ligação para o *peer* onde se encontra o ficheiro escolhido e é então efetuada a transferência do mesmo após confirmação por parte do requerente.

Transferência de Ficheiros

- Protocolo de Comunicação

Protocolo de Comunicação

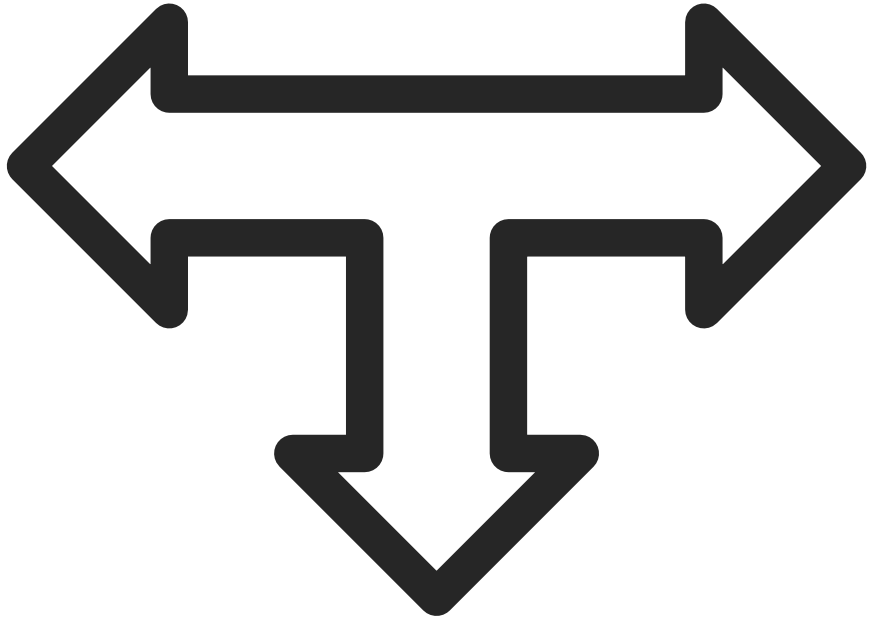
Antes de ser feita qualquer transferência, os *peers* comunicam entre si através de um protocolo de comunicação.

O protocolo de comunicação utilizado é uma adaptação do protocolo *KnockKnock* (c) Oracle .

Este protocolo consiste em confirmar junto com o *peer* que pede o ficheiro, se este confirma a receção do mesmo. Em caso afirmativo a transferência do ficheiro procede, em caso negativo a transferência é cancelada e o *peer* é informado que a transferência não irá decorrer.

Threads

- Integração
- *Threads* na aplicação



Integração

De forma a cada *peer* poder realizar tarefas de envio e receção em simultâneo são utilizadas *threads* que executam cada tarefa dessas paralelamente à execução do programa principal, ou seja, nenhuma outra tarefa é interrompida apenas porque é necessário obter a listagem de ficheiros (conceito de *multitasking*).

Com este tipo de implementação também é possível o envio/receção de vários ficheiros/listas em simultâneo recorrendo a uma *thread pool.*

Threads na aplicação

Durante a execução da aplicação existem *threads* que estão sempre em execução em *background*, estas *threads* são responsáveis por

- "Estar à escuta" de pedidos de envio de lista de ficheiros;

- Executar cada pedido de transferência de ficheiro;

- Encaminhar pedidos de transferência de ficheiros para serem executados por uma *thread pool*;

- Estar preparada para realizar a comunicação de forma a transferir um ficheiro;

Threads na aplicação

Para além das *threads* que estão em execução durante todo o ciclo de vida da aplicação, também são criadas *threads* para a realização de tarefas com execução finitas.

Tarefas tais como

- Transferência/Envio de um ficheiro;

- Transferência/Envio de uma lista de ficheiros;

- Pedido de ficheiro/lista de ficheiros;

Sockets

- *Multicast Socket*
- *Server Socket*
- *Socket*

Multicast socket

É utilizado um *multicast socket* por cada *peer* de modo a ligarem-se à rede onde poderão obter as informações sobre os outros *sockets* ligados à mesma, informação vital para futuras ações de troca de ficheiros.

Este *multicast socket* utiliza o protocolo UDP *(User Datagram Protocol)* , o que significa que não fornece nenhuma garantia que o pacote onde contém o pedido é recebido/entregue.

Por esta razão este *socket* apenas é usado para informar os outros *peers* que houveram pedidos de envio de lista de ficheiros.

Server socket

Para troca de informação que não pode ficar perdida na rede, é utilizado um *server socket* que assenta no protocolo TCP *(Transmission Control Protocol)* que assegura o envio dos dados.

Este *socket* é utilizado para

- Receber as listas parciais de cada *peer* da rede;
- Receber todos os pedidos de ficheiros;
- Envio de ficheiro;
- Utilização do protocolo de comunicação da aplicação;

Socket

Na outra extremidade da conexão existe um *socket* que se liga ao *server socket* previamente criado e é utilizado para troca de informação.

É utilizado para

- Envio da lista de ficheiros para um outro *peer*;

- Envio de um pedido de ficheiro;

- Receção de um ficheiro;

- Utilização do protocolo de comunicação da aplicação;

Prefácio

Aplicação criada no âmbito da unidade curricular de Sistemas Distribuídos utilizando os conceitos lecionados nas aulas durante o semestre.

Para a criação desta aplicação foi utilizada a linguagem Java e o IDE Netbeans.

O tema da aplicação assenta no principio de comunicação entre aplicações, através da rede, recorrendo a *sockets* e a protocolos de comunicação tais como UDP e TCP.