Instituto Superior de Engenharia de Coimbra

DEIS/LEI

Programação Distribuída

Trabalho Prático Fase 1: PD

João Baptista → a2020131684@isec.pt - LEI

Bruno Amado Sousa → a2020132971@isec.pt - LEI

Jorge Gabriel Querido dos Santos → a2020133143@isec.pt - LEI

Coimbra, 05 de dezembro de 2022

ÍNDICE

Conteúdo

1-Introdução

Este relatório pretende explicar os vários sistemas usados no trabalho prático de programação distribuída de 2022/2023. O objetivo é aprender a implementar os diferentes componentes que constituem uma aplicação servidor/cliente em java sockets.

2 – Arquitetura

O projeto está dividido em duas partes:

Cliente e Servidor.

O cliente é composto por uma máquina de estados que nos permite controlar as vistas do utilizador (JAVA FX). Composto por um ficheiro ClientManager que permite controlar o envio de mensagens tcp e udp quando necessário. Também é composto por um mini servidor tcp que recebe ack's do servidor tcp cujo este se encontra ligado.

O servidor, é composto por um ficheiro ServidorManager que é responsável por controlar o servidor TCP, o servidor UDP, Multicast – Receiver e Multicast – Sender.

Basicamente, o servidor aguarda pelo cliente no udp e distribui uma lista de servidores ativos. Isto acontece no ficheiro preClientUdp.

De seguida, é criada uma nova conexão ,com a ajuda da thread ServerTCP, que ficará responsável por receber os pedidos do utilizador.

E resumidamente, esta é a nossa arquitetura.

3 – Server – UDP

O servidor udp aguarda por clientes, que ao conectarem – se enviam um ficheiro serializado com a informação suficiente para registar ou fazer login.

De seguida, se estiverem autenticados, estes são reencaminhados para um dos servidores da lista com servidores tcp ativos, cuja é enviada no final do ficheiro udp.

4 – Server – Multicast

Existem vários ficheiros referentes ao multicast, no entanto apenas o receiver (Thread) recebe mensagens por multicast. Mensagens que são compostas por um objeto serializado que permite verificar o tipo desta (Heartbeat, Prepare, Commit, Abort…).

Para enviar HeartBeats é criado uma thread que em cada 10 segundos envia um heartbeat para o multicast. Estes heartbeats têm informação suficiente para os servidores se fazerem conhecer aos restantes ativos.

Se um servidor receber o seu heartbeat 4 vezes e um servidor não ter enviado um heartbeat, esse servidor será removido da lista de servidores ativos.

Além dos HeartBeats, temos Senders responsáveis para enviar Mensagens do tipo Prepare, Commit e Abort.

5 – Server – TCP

O servidor TCP baseia -se na arquitetura Client/Worker, onde basicamente o ServerClient (Ficheiro/Thread) irá adicionar novas ligações a uma lista de clientes permitindo assim conhecer a sua carga para disponibilizar aos restantes servidores.

Cada cliente terá uma ligação com o servidor com menos carga, cujo irá receber códigosTCP que permitem identificar a ação que o utilizador quererá realizar. Desta maneira é possível haver sincronicidade entre clientes e todos os servidores se manterão atualizados.

Além disto, o ServerTcp cria uma thread timer que irá enviar um ack para cada cliente para que seja verificado se este continua online ou se houve alguma falha na sua aplicação, se este for o caso remove -o da lista.

6-Base de dados

O servidor é responsável por fazer qualquer alteração na base de dados e obter dados para o cliente a partir dela. Estes processos acontecem no ficheiro DBmanager.

7 – Conclusão

Este trabalho permitiu aprender implementar diferentes tipos de servidores e criar aplicações do género Servidor/Cliente. Acreditamos com este trabalho conseguiremos no futuro implementar este tipo de aplicações com mais facilidade.