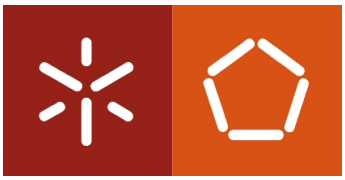


Universidade do Minho

Departamento de Informática

Relatório de Redes de Computadores
Grupo 59

Inês Alves (A81368) João Lopes (A80397)
João Gomes (A82428) Sofia Teixeira (A80624)

6 de Janeiro de 2019

Conteúdo

1 Introdução

No âmbito da Unidade Curricular de Sistemas Distribuídos, foi-nos proposta a implementação de um trabalho prático sobre Alocação de Servidores na Nuvem.

Assim, pretende-se desenvolver um sistema de compra de servidores. Esta compra pode ser feita a pedido ou através de um leilão. Se for uma reserva a pedido, o cliente escolhe o tipo de servidor que pretende adquirir. Após esta escolha, é lhe cobrado o preço relativo ao tempo utilizado. Se a reserva for feita através de um leilão, o cliente pode licitar por um servidor. Se ganhar o leilão o servidor é lhe concedido e é cobrado o preço nominal horário relativo ao tempo de utilização.

2 Implementação

2.1 Utilizador

A classe Utilizador permite representar qualquer tipo de utilizador do nosso sistema de Alocação de Servidores. Esta classe tem os atributos do utilizador: email, password, hostname e a sua dívida. Juntamente com os seus métodos de acesso e modificação.

2.2 Servidor

A classe Servidor representa o servidor que vai ser atribuído ao utilizador. Tem como atributos o seu id, tipo, o seu estado e o seu preço.

2.3 Reserva

A classe Reserva representa a atribuição de um servidor a um utilizador. Tem como atributos o Servidor que é atribuído a um Utilizador, que é também atributo, juntamente com a data de atribuição e também o identificador da reserva.

2.4 Handler

A classe Handler é a que efetua a comunicação entre a classe Menu e o utilizador, ou seja, fornece as opções existentes, o utilizador faz as suas escolhas, informa esta classe das escolhas realizadas e esta, por sua vez, comunica com a classe Menu, de modo a que as escolhas sejam validadas (ou não) e processadas.

2.5 Menu

A classe Menu, já referida em cima, é responsável por, recebendo um input proveniente do handler, realizar todas as operações lógicas de forma a garantir o funcionamento do programa.

2.6 DadosServidor

A classe DadosServidor é a classe onde são guardados os servidores existentes. Possui um Map de servidores onde a chave é o seu id e uma lista de servidores reservados. Permite-nos ainda saber que servidores se encontra, ou não, livres.

2.7 DadosCliente

A classe DadosCliente, analogamente à classe anterior, representa a classe onde são guardados os clientes que acedem ao sistema.

2.8 Cliente

A classe Cliente efetua a ligação deste com o servidor e inicia duas threads: uma que está associada à classe Ler e outra que está associada à classe Escreve.

2.9 Ler

Esta classe lê as respostas do servidor e imprime-as.

2.10 Escreve

Classe responsável por ler do *stdin* e escrever para o programa.

3 Comunicação Cliente-Servidor

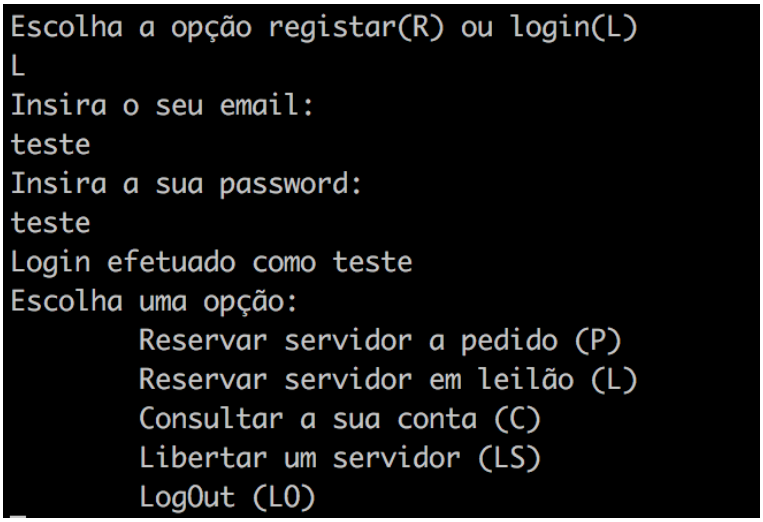
De modo a que os requisitos pedidos no enunciado do trabalho prático, tanto o Servidor como o Cliente foram implementados via Sockets (TCP). A classe "Handler"atua como o cliente, ao inicializar o programa é criada uma ServerSocket que é passada para cada um dos handlers criados (o número é escolhido ao inicializar o programa). Estes handlers irão lidar com os inputs do utilizador, permitir que este comunique com o handler e comunicar com o servidor o input do utilizador para este realizar todas as operações. A class "Menu"atua como o servidor, ao inicializar o programa é carregada uma base de dados para o "Menu"que depois, recebendo inputs do "Handler"irá permitir o correcto funcionamento do programa. Após realizar todas as operações necessárias devolve para o "Handler"um código que simboliza o resultado da operação.

4 Controlo de Concorrência

De forma a garantir o controlo de concorrência utilizamos locks (mais especificamente ReentrantLocks). Estes locks são colocados nos métodos em que é necessário garantir que não estão a ser acedidos por várias threads ao mesmo tempo, tais como, o método de reserva de um servidor a pedido, o método de reserva de um servidor em leilão, etc. Deste forma conseguimos garantir que tudo corre normalmente não havendo conflitos entre as operações das threads.

5 Interface Gráfica

A nossa interface gráfica é visualmente muito simples, já que o propósito deste trabalho era ser utilizado na linha de comandos.



6 Conclusão

Em suma, neste trabalho foram aplicadas as noções dadas nas aulas sobre concorrência, utilização de threads e de servidores. Foram aplicados locks para garantir o controlo de concorrência. Uma vez que a implementação de threads é uma das bases deste projeto é crucial garantir o controlo de concorrência. No entanto, esta implementação requer algum cuidado no que toca ao acesso aos dados, já que não queríamos que, por exemplo, 2 utilizadores pudessem licitar ao mesmo tempo.

Foram ainda utilizados sockets para a criação dos servidores e para a conexão dos utilizadores.

Para finalizar, o programa funciona como era esperado e implementa todas as funções pedidas no enunciado.