Universidade do Minho
Mestrado Integrado em Engenharia Informática
Sistemas Distribuídos

Trabalho Prático

Grupo 29
André Vieira - A78322
Eduardo Rocha - A77048
Ricardo Neves - A78764
6 de Janeiro de 2019

Conteúdo

1 Introdução

Neste trabalho, realizado no âmbito da Unidade Curricular de Sistemas Distribuídos, foi-nos pedido que desenvolvêssemos um Sistema Distribuído capaz de gerir a reserva de *Cloud Servers*.

Estes servidores *Cloud* estariam divididos entre categorias e deveria ser possível aos utilizador escolher entre tentar reservar os servidores normalmente ou através de um leilão. Nestes leilões, os utilizadores competem entre si para ver qual utilizador está disposto a pagar mais para reservar o servidor para seu uso. Porém, as reservas por leilão podem ser canceladas caso não haja mais servidores disponíveis nessa categoria para uso de reserva a pedido.

2 Funcionalidades do Sistema

Este sistema possui todas as funcionalidades detalhadas no enunciado do trabalho.

O servidor é capaz de autenticar todos os utilizadores a que ele se ligam, quer seja através de *login* para uma conta existente ou registo de nova conta.

Após o cliente estar autenticado, é-lhe permitido verificar a dívida acumulada por ele, reservar novos servidores (a pedido ou a leilão), cancelar as suas reservas atuais e ver o estado de todos os servidores existentes no sistema.

3 Elementos do Sistema

Iremos agora detalhar quais as classes constituintes do nosso sistema. Estas são:

- **Main** - inicia o Servidor ou o Cliente dependendo da opção do utilizador
- **Servidor** - estabelece conexões com os Clientes que se tentam ligar ao sistema e cria novos *Workers* que irão tratar dessas conexões
- **Worker** - tem as funções que constituem as funcionalidades do sistema; cria uma nova *Thread* que verifica se o seu cliente perdeu alguma reserva
- **Cliente** - permite ao utilizador comunicar com o sistema
- **ReaderCliente** - lê as linhas enviadas ao Cliente e imprime-as para o utilizador
- **CloudServer** - define a constituição de cada *Cloud Server* incluído no sistema
- **Conta** - define a constituição das contas de todos os Clientes
- **AuctionController** - são iniciados pelos *Workers* quando necessário, controlam os leilões que decorrem no sistema

4 Dados de Teste

O funcionamento do nosso sistema foi assegurado com a implementação de dados de teste capazes de suportar todas as funcionalidades do sistema.

O grupo adicionou vários Servidores *Cloud*, distribuídos entre três categorias: *large*, *medium* e *micro*. Adicionamos também várias contas de Clientes e atribuímos a duas destas contas um Servidor *Cloud*, sendo uma destas reservas feita por pedido e a outra por leilão. O objetivo da inclusão destes dados de teste foi poder experimentar todas as funcionalidades implementadas imediatamente, incluindo as opções de substituir e eliminar reservas.

5 Estruturas de Dados Usadas

Para armazenar todas as informações relativas aos *CloudServers*, usamos um *HashMap* com vários *HashMaps* dentro de si. Os vários *HashMaps* interiores estão identificados por uma categoria. Cada elemento dessa categoria é um *CloudServer*.

As contas dos Clientes estão armazenadas dentro de um único *HashMap*, em que cada conta é identificada pelo *email* usado na sua criação.

Usamos um *ArrayList* para guardar informação relativa a reservas por leilão perdidas quando uma reserva a pedido a elimina.

As contas dos Clientes têm dentro de si um *HashMap* que guarda as reservas feitas por esse utilizador.

Finalmente, os *CloudServers* usam um *HashMap* para guardar as ofertas feitas por participantes em leilões.

6 Comunicação

A comunicação entre os Clientes e o Servidor é feita inicialmente através de um *Socket* no lado do Cliente e um *ServerSocket* no lado do Servidor.

O Servidor cria um *ServerSocket* para receber conexões novas vindas de Clientes e, para cada uma, cria um novo *Socket*, que será gerido por uma instância da classe *Worker*. Como requisito do programa, cada *Worker* comunica apenas com o Cliente que lhe foi atribuído, ou seja, só usa uma *Socket*.

A comunicação é orientada à linha. Cada vez que o sistema envia ou recebe uma mensagem, envia ou lê sempre uma linha inteira. Para o envio de mensagens é usado um *PrintWriter* e para receber mensagens é usado um *BufferedReader*.

7 Controlo de Concorrência

Para controlar a concorrência do sistema e assegurar que as *threads* não trabalham com dados desatualizados usamos *locks* com o objetivo de impedir que certos dados sejam acedidos e possivelmente alterados quando uma *thread* os está a usar.

Um exemplo disto em ação encontra-se no processo de autenticação de utilizadores. Quando um Cliente quer ver se pode registar-se com um certo *email* terá que ver se esse já existe numa conta. Caso duas *threads* vejam se o mesmo *email* existe ao mesmo tempo poderemos ter uma situação em que uma *thread* insere um novo utilizador com esse *email* e logo de seguida outra *thread* introduz o seu cliente com esse *email*, apagando o primeiro utilizador. Para prevenir isso, impedimos a segunda *thread* de operar enquanto a primeira ainda não cedeu o seu *lock* sobre os dados relevantes.

8 Conclusão

Em conclusão, o grupo encontra-se satisfeito com o trabalho realizado. No ano anterior o grupo não foi capaz de concluir o Trabalho Prático com sucesso por isso ser capaz de o concluir este ano, tendo implementado todas as funcionalidades pedidas, é um resultado positivo para nós. Adicionalmente, achamos o trabalho uma ótima maneira de treinar os aspetos práticos assim como os teóricos da Unidade Curricular, o que nos ajuda a preparar para o resto da disciplina.

Porém, existem áreas do trabalho com que o grupo não se encontra tão satisfeito. Existem áreas do controlo de concorrência em que não sabíamos ao certo qual o tipo de controlo a usar, logo receamos que hajam problemas de controlo devido a isso. Adicionalmente, a formatação do código na classe Worker leva a que o código possa ser difícil de interpretar. Gostaríamos de ter podido formatar o código de forma a ser mais legível.

Apesar disto, achamos que concluímos satisfatoriamente o trabalho e que foi uma mais valia na completação desta Unidade Curricular.