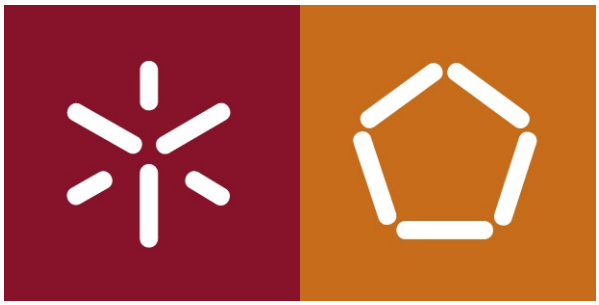

Universidade do Minho

Mestrado Integrado em Engenharia Informática

SERVER_BUY APP

SISTEMAS DISTRIBUÍDOS

João de Macedo (76268)
Nelson Gonçalves (78173)
Paulo Pacheco (77808)
André Salgueiro (77617)

Janeiro 2019

Conteúdo

1 Introdução

Este relatório tem como objectivo a apresentação do desenvolvimento deste projecto e das decisões tomadas durante a sua realização. O projecto consiste numa aplicação distribuída em que permite reservar e usar servidores virtuais para processamento e armazenamento. A aplicação desenvolvida deve ser capaz de efectuar requisitos tais como, a autenticação e registo de um Client dado um *e-mail* e uma *password*, podendo após a autenticação fazer a reserva a pedido ou a reserva de uma instância em leilão.

2 Estrutura do Relatório

A funcionalidade desta aplicação é constituída por várias fases, tendo inicio na ligação entre o Client e a Cloud. Após esta ligação, o Cliente é lhe permitido várias funcionalidades que são geridas através de uma gestão de threads feita pela Cloud.

Esta gestão de threads e toda a estrutura necessária para a sua funcionalidade será explicada detalhadamente nas seguintes sub-secções.(ver esta estrutura no final).

2.1 Client - Cloud

Quando o Client inicia a aplicação é estabelecida uma conexão entre o Client e a Cloud (que é o servidor da aplicação) através de sockets TCP. Quando a Cloud recebe essa conexão ela vai lançar uma thread, ServerWorker, que ficará responsável pela conexão com esse Client.

Quando a thread é lançada ela permite que o Client efetue o registo, no caso de não se encontrar registado, ou efetue o login no sistema. Após o login ser efetuado, a thread ServerWorker ficará encarregue de receber os pedidos do cliente e tratar desses mesmos pedidos. Esta thread apresenta um estado partilhado, o objeto Cloud, que contém todos os dados do sistema.

Para além desta thread outras threads são lançadas ao longo do programa, que são responsáveis por tratar de licitações e compras de servidores, quando estes não estão disponíveis. Estas duas threads serão apresentadas com maior detalhe posteriormente.

2.2 Reserva de um servidor a pedido

Para fazer a reserva de um servidor a pedido, o Client envia através do TCP socket essa intenção à sua thread ServerWorker, que por sua vez lhe dispõe ao Client os vários tipos de Server. O Client escolhe o tipo de Server que pretende e independentemente do tipo que escolhe, o ServerWorker vai ver se há algum Server desse tipo com a *flag* estado a "livre"primeiramente e depois se há com o estado "leilao". Caso exista um com o estado a livre, esse Server é adicionado à lista de Server do Client e o estado alterado para ocupado. Se o Server encontrado disponível tem o estado "leilao", é retirado o Server ao atual Client, sendo este notificado, e inserido na lista de Server do Client que reservou a pedido, alterando o estado para "ocupado".

Se após a verificação não existir algum Server disponível, tem de se acumular as propostas feitas pelos diversos Client. Para isso é criada uma thread para que a proposta de reserva a pedido do cliente fique em *standby*, sendo esta de volta ativa quando houver um Server disponível pela libertação de um Server por um outro Client. Quando a thread fica ativa,

adiciona o Server ao Client que fez a proposta e é lhe notificado que adquiriu um novo servidor na área das notificações.

Esta habilidade de criar uma thread para a proposta ficar *standby* é feita para que a ligação entre o Client e o ServerWorker não fique *stanby*. Se esta ficasse de tal modo, o Client não conseguiria fazer mais nada enquanto um Server não fosse libertado.

2.3 Reservar uma instância em leilão

No caso da reserva de uma instância de leilão, o procedimento é semelhante ao da reserva a pedido. O Client ao indicar o tipo Server que pretende licitar, vai indicar o valor que quer licitar e este valor inserido, vai ser testado pelo ServerWorker por este ter que ser menor que o valor do Server quando é reservado a pedido. Após esta verificação o ServerWorker vai verificar se há Server desse tipo com o estado a "livre", em que caso tenha, o Client adquire o Server e a *flag* estado é atualizada para "leilao". Caso contrário, procede-se como na reserva a pedido. É criada uma thread para ficar em *standby* de modo a que se possa acumular licitações e o Client possa usufruir da aplicação apesar de ter a licitação em *stanby*.

Se um Server for libertado, e não haja nenhum Client em *standby* para a reserva a pedido desse tipo de Server, vai-se verificar qual das licitações acumuladas para esse tipo é a maior. A thread da licitação maior é ativa e o Client que a obtém, adquire o Server, alterando o estado para "leilao"e a sua licitação é eliminada, notificando o Client que adquiriu o Server.

3 Estrutura Geral

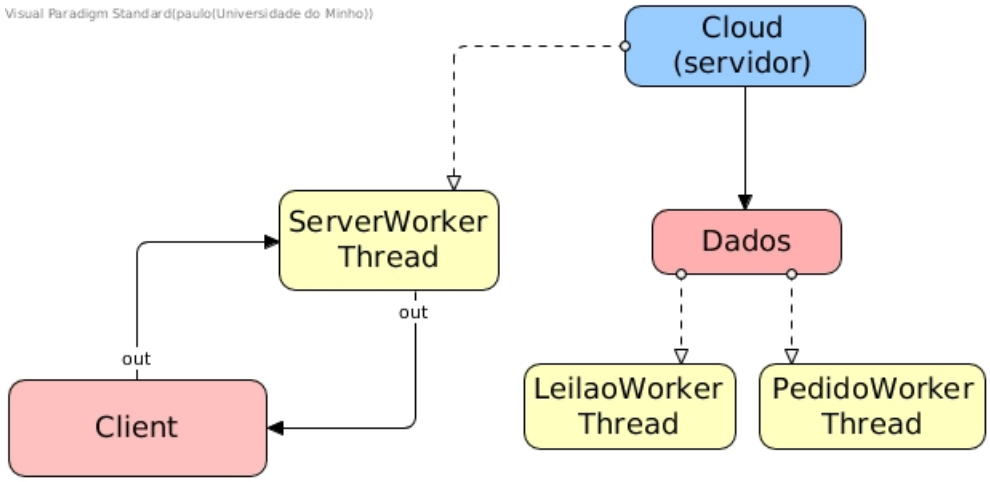

Figura 1: *Estrutura geral do sistema*

- **Dados:** Representa a classe que contém toda a informação do sistema. Apresenta coleções de dados que armazenam todos os Clientes do sistema, todos os servidores existentes de cada tipo, uma lista com todos os Clientes que efetuaram pedidos de um dado tipo de servidor, a lista de licitações associados a cada tipo de servidor, uma lista de nickname do Client e Condition associada de todos os clientes á espera de servidores a pedido, e por último, uma lista semelhante mas aplicada aos servidores a leilão.

A razão de termos coleções de dados que armazenam uma Condition diferente a Clientes diferentes foi evitar o uso do signalAll(). Em vez de adormecer todos as threads numa mesma condição e acordar todas de cada vez fazendo a verificação do ciclo while, lançamos uma thread que fica adormecida e adicionamos o nick do cliente bem como uma condição nova a si associada à hashtable. Deste modo, as verificações que estariam associadas ao ciclo while são agora feitas pelo servidor acordando uma só thread, sendo esta a que vai adquirir o Server. Deste modo é possível diminuir a espera ativa, poupando assim os recursos.

4 Funcionalidades

Aquando a implementação desta aplicação foram necessárias desenvolver várias funcionalidades básicas no que toca à interface com o Client, permitindo tanto o registo e autenticação deste, como várias outras funcionalidades essenciais respetivas à reserva de um servidor.

4.1 Registo e autenticação de um Client

Inicialmente é dado a escolher ao Client qual o tipo de ação que pretende selecionar, a de efetuar um registo ou a de efetuar login. Após efetuado o login, são apresentadas várias funcionalidades ao Client:

- *Comprar* Servidor
- *Licitar Servidor*
- *Gerir Servidores e Licitações*
- *Dados pessoais*
- *Notificações*
- *Logout*

4.2 Reserva de um servidor a pedido

Para reservar um servidor a pedido, o Client seleciona a opção *"Comprar Servidor"* e de seguida são apresentados os vários tipos de servidores para o Client escolher o tipo de servidor que pretende reservar. Caso não hajam servidores disponíveis nem servidores reservados a leilão, é lançada a *thread PedidoWorker* que vai ficar *stanby* à espera que algum servidor fique disponível. Quando houver algum Server disponível a *thread* "acorda"e é atribuído o Server disponível ao respetivo Client.

4.3 Reserva de um servidor em leilão

Caso o Client pretenda reservar um servidor por um preço menor que o preço nominal fixo, seleciona a opção *"Licitar Servidor"* e de seguida são apresentados os vários tipos de servidores para o Client escolher o tipo de servidor que pretende reservar. Ao escolher o tipo pretendido, o Client tem de indicar o preço que pretende propor. Se houverem servidores do respetivo tipo disponíveis é automaticamente atribuído o servidor ao Client. Caso contrário a *thread LeilaoWorker* "adormecer"até que seja atribuído um servidor a esta licitação.

4.4 Libertar um servidor e cancelar licitações

Quando o Client pretender libertar um servidor, escolhe a opção *"Gerir Servidores e Licitações "* , de seguida através da opção *"Ver Servidores"* são apresentados os servidores que o Client tem reservados e por fim indica qual o id e tipo do servidor que pretende libertar.

Por outro lado, se no *"Menu Gerir"* o Client selecione a opção *"Ver Licitações"*, são apresentadas todas as licitações que estão a aguardar servidores disponíveis, tendo o Client a possibilidade de cancelar qualquer licitação efetuada.

4.5 Consultar a conta corrente

Um Client autenticado tem a possibilidade de consultar o valor em dívida de acordo com a utilização dos recursos da aplicação. Para tal, no *Menu Principal*, selecionando a opção *Dados Pessoais* é apresentada a dívida do respetivo Client, tendo este a opção de abater a dívida ou parte dela.

4.6 Notificações

Esta opção é necessária para que o ServerWorker consiga avisar o Client caso perca algum Server ou adquira um.

5 Conclusão

Após implementada a nossa aplicação, e analisado o resultado obtido, podemos afirmar que, de uma forma geral, achamos a abordagem que seguimos adequada. Todas os requisitos inicialmente proposto para este projeto foram respeitados e implementados com sucesso.

Com o objetivo de armazenar os dados da aplicação, implementamos um sistema de backup que sempre que é realizada uma funcionalidade é armazenado num ficheiro os dados atuais da classe Dados, o que permite diminuir possíveis perdas de dados.

Por fim, concluímos que a realização deste projeto foi implementada com relativo sucesso. Contudo, temos consciência que existem aspetos a melhorar, nomeadamente a implementação de uma Base de Dados e a utilização de uma interface gráfica.