Escola de Engenharia
Departamento de Informática

Mestrado Integrado em Engenharia Informática

Trabalho Prático

Sistemas Distribuídos

Gestor de Leilões

Filipe Daniel Mota 62153
Norberto João Sobral 60982
Rafaela Guerra 60991

Introdução

Foi proposto aos alunos a realização de um trabalho cujo objectivo é o desenvolvimento de uma aplicação que permita a gestão de um serviço de leilões.

Durante o desenvolvimento desta aplicação deparamo-nos com alguns problemas, sendo preponderante destacar a concorrência nos acessos aos recursos do sistema.

De acordo com o que nos foi pedido, a nossa aplicação deve aplicar algumas restrições no acesso a algumas funcionalidades, neste caso, restringir o acesso aos dois tipos de utilizador de forma a que cada utilizador apenas tenha acesso às suas funcionalidades, isto implica alguns cuidados, nomeadamente a necessidade de garantir que sempre que um utilizador pretenda aceder a uma determinada funcionalidade esse acesso seja garantido sem que hajam situações de interbloqueio. Desta forma, no desenvolvimento desta aplicação utilizamos mecanismos de exclusão mútua e variáveis de condição. Conforme nos foi sugerido no enunciado do projecto, recorremos também o uso de sockets para estabelecer a comunicação entre servidor e clientes.

Desenvolvimento

Gestor

A nossa aplicação tem uma classe Gestor que é directamente recebida por todas as threads que são criadas excepto a *Thread_LeCliente*. Esta classe contém uma listagem dos leilões em curso e terminados assim como o histórico de licitações. Contém também uma lista de todos os vendedores e compradores associados ao nosso sistema. Existem também duas *locks* para que seja possível mapear concorrencialmente as licitações e os leilões. É nesta classe que estão os métodos relacionados com Leilões e Licitações, vamos fazer uma pequena descrição desses métodos, referindo a negrito os métodos que contêm ***locks()*** e ***unlocks()***. Este mecanismo serve para permitir que os acessos aos leilões e licitações sejam feitos de forma concorrente, desta forma, no início de cada operação é feito um ***lock(***) e no fim um ***unlock()*** :

constroi_Leilao - Cria um novo leilão.
inicia_Leilao - Inicia um novo leilão.
termina_Leilao - Finaliza um leilão que está em curso.
constroi_Licitacao - Cria uma licitação.
fazLicitacao - Efectua uma licitação.
mudaValor - Altera o valor actual de um leilão após haver uma licitação superior.
listaEmCurso - Lista todos os leilões que estão a decorrer.
listaTerminados - Lista todos os leilões terminados.
listaMeusLeiloes - Lista os leilões de um vendedor.
listaMinhasLicitacoes - Lista o histórico de licitações de um comprador.

Leilão

A nossa classe Leilao é constituida por um Int que será o id do leilão, duas Strings que correspondem à descrição e ao nome do vendedor, um Boolean que corresponde ao estado do leilao onde interessa saber se está em curso ou terminado, e dois Floats que correspondem aos valores actuais e iniciais para as licitações.

Licitação

A classe Licitacao é constituída por um Int que corresponde ao id do leilão a qual a licitação será associada, uma String que corresponde ao nome do comprador/licitador e um Float que corresponde ao valor que é usado para a licitação.

Utilizadores

A nossa aplicação tem uma classe Utilizador que contém campos que irão ser comuns nos dois tipos de utilizadores suportados pelo nosso sistema, esta classe é composta por duas Strings que correspondem à informação de login dos utilizadores (nome e password).

Decidimos criar duas novas classes que fazem extend à classe Utilizador de forma a diferenciar o tipo de cada utilizador pois contêm variáveis diferentes, essas classes são a Vendedor e Comprador.

A classe Vendedor contém um Int que lhe faz associar um tipo, assim é mais fácil para o sistema identificar com que tipo de utilizador está a trabalhar, e contém também uma lista de itens leiloados associados a cada vendedor.

A classe comprador contém também um Int identificador do tipo de utilizador e uma lista de leilões onde efectuou licitações.

Cliente/Servidor

Cliente

Como sugerido no enunciado, cada cliente estabelece uma ligação através de um socket definido no servidor. Existe uma troca de informação entre estes dois na medida em que cada pedido lido na consola Cliente é enviado para o Servidor e quando um pedido é concluído o cliente recebe uma resposta.

Para cada cliente é criada uma thread (*Thread_LeCliente*) que lê as mensagens do servidor, assim conseguimos garantir que um novo pedido não fique pendente de uma resposta a um pedido anterior.

Servidor

O Servidor contém sockets, um Gestor que é a nossa main classe onde estão incluídos os leilões em curso e terminados assim como histórico de licitações e os dois tipos de utilizadores suportados pelo sistema (Vendedores e Compradores).

Aqui, o socket é criado de forma a possibilitar a ligação entre cliente e servidor.

Quando há uma nova conexão com um Cliente é criada uma nova thread (*Thread_Cliente*), esta thread tem como função receber os pedidos de validação de utilizadores assim como receber os seus pedidos após a autenticação.

A cada pedido recebido do cliente é criada uma nova thread (*Thread_Comandos*), esta thread verifica os pedidos recebidos e executa no Gestor, desta forma garantimos que não existem pedidos pendentes de conclusão de pedidos anteriores, assegurando também que existe concorrência no Gestor. Como já foi referido, quando um pedido é concluído é enviada uma resposta ao cliente.

É também criada uma thread (*Thread_Consola*) que permite a utilização da consola do servidor.

Manual de utilização

Login/Logout

De forma a aceder ao sistema o utilizador terá que efectuar e autenticar o logic, usando o seguinte comando:

- *Login:username:password*

O utilizador pode efectuar logout a qualquer momento usando o comando:

- *Logout*

Iniciar Leilão

Um utilizador vendedor tem permissões para iniciar um leilão, usando o seguinte comando:

- *Leilao:NomeDoLeilão:ValorInicial*

Quando o vendedor cria um leilão, o servidor devolve o id do leilão.

Listar

Quer o utilizador vendedor, quer o utilizador comprador, podem a qualquer momento ver os leilões a decorrer usando o seguinte comando:

- *MinhasLicitacoes*
- *MeusLeiloes*

Aos utilizadores vendedores é devolvida a lista de leilões e um marcador '*' nas vendas pertencentes aos próprios. Aos utilizadores compradores, caso tenham a licitação mais alta, num ou mais leilões, estas aparecem marcados com um '+'.

Licitar um item

Só o utilizador comprador pode licitar um, ou mais itens, usando o comando:

- *Licitacao:IdDoLeilão:Valor*

Terminar Leilão

O utilizador vendedor pode a qualquer momento terminar um leilão usando o seguinte comando:

- Termina:IdDoLeilão

O sistema notifica os utilizadores com ofertas, quem ganhou a licitação, e o valor final.

Conclusão

Depois de fazermos testes com vários clientes e o servidor não detectamos anomalias no processo de acesso simultâneo a recursos do nosso sistema, assim sendo, concluímos que conseguimos alcançar uma solução credível para o problema identificado na concepção do projecto.

Temos conscîencia que a solução encontrada terá sempre de ser adaptada a problemas do “mundo real” e que cada sistema é independente nas suas necessidades podendo ser necessário mudar a forma abordagem.