UNIVERSIDADE
DE ÉVORA

Sistemas Operativos II

Primeiro Trabalho:

Sistema de Partilha de Disponibilidades e Necessidades

Ano lectivo 2019/2020

Ana Silvério nº 37561

João Queimado nº 38176

Situação Problema

Uma situação pretende informar a população da existência de produtos provenientes da loja X, ou da necessidade crítica de um produto em específico por parte de uma ou mais pessoas. Permitindo a um utilizador informar o sistema de que encontrou o produto na loja Y, de modo a que o sistema possa cominucar aos interessados qual a loja onde se encontra o produto solicitado.
No entanto, o utilizador não tem conhecimento de quem solicitou o pedido, por questões de privacidade.

A implementação da solução passa por implementar uma aplicação servidor, que terá os dados em armazenamento persistente, e a uma aplicação cliente, com funcionalidades de consulta de necessidades, reporte disponibilidade de um produto num local (loja) e um registo de necessidade do produto Z.
Caso ocorram falhas de comunicação com o servidor, a aplicação cliente deve reter os últimos dados de forma persistente, tanto o que estava a mostrar ao utilizador como os elementos que iria enviar ao servidor, para que não se percam em caso de crash ou reboot local.
De cada vez que é registada uma necessidade, a aplicação do utilizador deve receber um código de registo único. O sistema deve ter forma de distinguir os utilizadores. Quaisquer parâmetros de configuração devem estar fora do código, sendo passados como argumento à aplicação ou lidos de um ficheiro de propriedades (ver java.util.Properties).

Implementação: Servidor

- ***DBManager Class***

  Tal como o nome indica, é a classe de gestão da Base de Dados do sistema. É onde se fica tudo o que tenha a haver com acessos/consulta da base de dados. As queries em SQL permitem saber se o produto pedido/invocado pelo cliente/utilizador da aplicação já se encontra na base de dados, tal como se um certo utilizador X já tem "ficha" na base de dados. Tem-se uma gestão dos produtos, dos utilizadorores/clientes e das "lojas".
  A classse *DBManager* vai abstrair toda a base de dados para todo o sistema.

- ***LoginAgentImp Class***

  Esta é a Classe correspondente à aplicação Utilizador. O utilizador estabelece contacto com o sistema e atuará como cliente. Assim, qualquer utilizador que se ligue à aplicação cliente será inserido na base de dados e/ou está em estado de *logged in* no sistema.
  A classe que define a estrutura do objecto remoto *LoginAgent*.

- ***ProductAgentImp Class***

  Classe responsável pelos produtos. Quer isto dizer, nesta classe encontram-se definidas as operações de gestão de produtos que podem ser realizadas sobre estes.
  Classe que implementa a interface remota *ProductAgent*, com os seus respetivos metodos remotos.

- ***RMIController Class***

  Classe que trata do *RMI registry*, serviço de nomes do Java RMI. Ou seja, classe que tem como função registar objetos remotos no serviço de nomes do java RMI.

- ***RequestAgentImp Class***

  Classe do género da classe ProductAgentImp, partilham do mesmo conceitos. Nesta classe encontram-se a gestão dos pedidos. Quer isto dizer, quando o cliente requisita um produto ao sistema, este reserva-o. Assim que o produto requisitado se encontrar disponivel, o sistema informa o cliente.
  Classe que implementa a interface remota *RequestAgent*, com os seus respetivos metodos remotos.

- ***Servidor Class***

  É a classe "main" do sistema, do lado do servidor. Nesta classe, o servidor dá inicio à base de dados e vai criar as instancias das classes remotas e adiciona-as ao *RMIController*, sistema de nomes. Só tem cocnhecimento de que um utilizador está *logged in* e qual o seu nome.

Implementação: Interface Remota

- ***LoginAgent Class***

  Interface remota do objeto remoto, *LoginAgent.*

- ***ProductAgent Class***

  Interface remota do objeto remoto, *ProductAgent.*

- ***RequestAgent Class***

  Interface remota do objeto remoto, *RequestAgent.*

Implementação: Cliente

- ***Client Class***

  A classe referente à aplicação cliente.

  É a classe que permite que o utilizador estabeleça contacto com o sistema.

- ***CommantExec Class***

  Nesta classe encontram se todas as interações do utilizador com o sistema. O utilizador pode fazer *log in* e *exist* no sistema, adicionar produtos (add(produto)), consultar pedidos feitos por este (show()) e adicionar o local onde se encontram os produtos (set(produto, local)) nas tabelas da base de dados através de um conjunto de métodos e propriedades.

- ***RemoteManager Class***

  Classe que funciona como que o *Back end* da classe anterior, que será o *front end*. Nesta classe ocorrem as operações referidas anteriormente, a um nivel interno no sistema. Mais especificado e detalhado, a que o utilizador não acede diretamente.

- ***Storage Class***

  Classe responsável por reter em memória os últimos dados de forma persistente, última mensagem enviada e recebida, do *RemoteManager*.

Observações

- ***Observações Positivas***

Os utilizadores são distinguidos entre si por um nome, ou seja, cada utilizador insere no sistema o seu nome para *log in.* Esse nome é registado no processo de log in, e se o nome não existir na base de dados então o sistema insere-o na base de dados, criando o log in.
- Se o utilizador pretender um produto que esteja em sistema e não tiver uma localização física, o sistema irá criar um pedido que associará o utilizador ao produto e atribuir-lhe-á um código.
- Se o utilizador pretender um produto que esteja em sistema e que tenha uma localização física, o sistema irá notificar o utilizador do local onde se encontra o produto pretendido, sem a criação de um pedido.
- Se o utilizador pretender um produto que não esteja em sistema, então o sistema vai criar na base de dados o produto e o pedido para esse produto.

O sistema permite consultar todos os pedidos feitos por um utilizador.
Se um utilizador encontrar a localização física de um produto, o sistema permite que esse utilizador possa inserir a localização encontrada para esse produto.
Os utilizadores interagem com o sistemas através de alguns comandos:
- add product: quando o utilizador insere um produto no sistema pode ocorrer uma das tres situações se descritas anteriormente;
- set product localization: Criação da localização física para um produto.
- show product: Mostra todos os pedidos que um cliente, utilizador do sistema, tem em seu nome.

- ***Observações Negativas***

A classe storage não ficou concluida nem funcional.
O comando para os utilizadores mstorage não faz o que é suposto.
O sistema de notificação automático não teve tempo de ser implementado de modo funcional.