

**Sistemas de Distribuidos**

**Trabalho prático 1**

|  |  |
| --- | --- |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Autores | Daniel Abreu nº 38539  João Pereira nº 37628 |
|  | Luís Lucas nº 38849 |

Semestre de Verão 2015/2016

Maio de 2015

**Índice**

[1. Introdução 1](#_Toc417484093)

[2. Formulação do Problema 3](#_Toc417484099)

# 

# Introdução

# Desenvolvimento

Podemos separar a solução em duas grandes partes, o cliente que no contexto do problema será o projecto *Super*, este projecto é uma aplicação de consulta de stocks de produtos de supermercados, para que esta consulta seja abrangente a todos os supermercados de uma hipotética cadeia de supermercados esta aplicação terá que comunicar com um servidor que lhe transmitirá essa informação. O projecto *Zone* reflecte o comportamento do servidor, este projecto tem a capacidade de poder comunicar com outros servidores de forma a ter conhecimento do stock global de todos os supermercados registados nos servidores de zona. Como este servidor realiza pedidos a outros servidores obtendo informação do stock disponível, e fornece essa mesma informação ao seu cliente, fazendo deste componente um *peer-to-peer*.

De forma a existir um contrato conhecido e respeitado entre estas duas entidades foi também criado um projecto que é uma biblioteca *.DLL* necessária na compilação do projecto *Zone* como do projecto *Super*, neste projecto estão definidas as interfaces *IZone* e *IStockManager* que serão as entidades necessárias de conhecimento global para a implementação da solução adoptada.

public interface IStockManager

{

IZone zone { get; set; }

IEnumerable<Item> GetStock(string it);

}

public interface IZone

{

void Register(IStockManager stockManager, IEnumerable<Item> stock);

void Unregister(IStockManager stockManager);

IEnumerable<Item> GetItemStock(string item);

}

A interface *IZone* tem um método *Register* que permite a um *Super* registar um objecto do tipo *IStockManager* e o stock que esse *Super* tem para que a aplicação global constituída por todos os servidores de zona tenha conhecimento do stock que cada super. Cada servidor zona comunica com um outro servidor de zona obtendo informação e enviando informação, no momento do registo de um *Super* numa zona esta comunica com a zona que esta conhece registando também o objecto IStockManager que recebeu assim como o stock obtido do *Super* que se registou nela própria. Este procedimento permite que a próxima zona se registe na sua zona conhecida, e a próxima zona numa outra e este procedimento irá continuar até que uma zona feche um círculo registando na zona inicial onde o *Super* se registou inicialmente. Desta forma todas as zonas têm conhecimento de todos os objectos *IStockManager* presente no sistema, assim como o stock de todos os *Super* registados.

Também presente na interface *IZone* está um método *Unregister* que recebe um objecto *IStockManager* eliminando-o da sua zona assim como o seu stock, este método tem um comportamento idêntico ao *Register*, que propaga a chamada a este método a sua zona conhecida eliminando o objecto *IStockManager* totalmente do sistema.

O método *GetItemStock* está também presente na interface *IZone* este método recebe o nome do produto que se quer saber o stock e retorna todo o stock desse mesmo produto presente no sistema.

Convém referir que o objecto *Zone* que implementa *IZone* é registado pelo servidor *Zone* em modo *Singleton* ou seja, cada vez que é obtida uma instancia deste tipo remotamente será sempre a mesma instância que será partilhada por todos os *Super* que se queiram registar no servidor, assim como a zona que tenha uma instância de outra, esta instância será a mesma que os *Super* obtêm.

A interface *IStockManger* tem uma porpriedade *IZone* esta propriedade permite que o objecto que implemente IStockManager tenha uma referência entre para *IZone*, esta propriedade permite ao *IStockManager* obter o stock de um produto chamando o método *GetItemStock* presente em *IZone*.

O método *GetStock* pertence a interface *IStockManager* este método tem como propósito obter o stock de um produto, a implementação deste método é feito ás custas do método *GetItemStock* da zona em que o *IStockManager* está registado.

Quando o servidor *Zone* fornece o objecto *StockManager* que implementa *IStockManager* fornece uma nova instância sempre que é solicitada uma instância de *IStockManager* para que isto seja possível este objecto é registado em modo *SingleCall*, esta decisão foi tomada tendo em conta que existe um *IStockManager* por cada *Super,* desta forma convém que exista uma instância de *IStockManager* cada vez que existe um registo de um *Super* num servidor de zona.

# Conclusão