ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO DE LEIRIA
INSTITUTO POLITÉCNICO DE LEIRIA

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Sistemas Distribuídos e Paralelos

2005/2006

**Engenharia Informática**

2.º ANO regime Diurno 3.º ANO regime Nocturno

**Engenharia Informática e Comunicações**

2.º ANO

# Buscas

Relatório da implementação do serviço *Buscas* do sistema de *Buscas Distribuídas*.

**Documento elaborado por:**
Sérgio Miguel Neves Lopes – aluno n.º 10635

**Versão 1.0**

Dezembro/2005

# ÍNDICE

# 1 Introdução

## 1.1 Descrição Geral do Trabalho

O BUD – Buscas Distribuídas, pretende ser um sistema distribuído com a capacidade para, regularmente, efectuar a pesquisa de ficheiros no parque informático de uma instituição.

### 1.1.1 Descrição da Segunda Etapa – Implementação do *Buscas*

Esta segunda etapa tem por objectivo a implementação do serviço de pesquisa denominado *Buscas*.
O objectivo é a implementação de um servidor híbrido permita a pesquisa de ficheiros e pastas nas diferentes máquinas.

### 1.1.2 Nota de Alterações à Primeira Etapa

Foram feitas algumas alterações ao código do servidor *Securitas* que, embora não alterem o protocolo de comunicação, podem impedir a correcta comunicação dos dois servidores implementados.
Foi incluído, nesta segunda etapa, o código fonte do servidor *Securitas* devidamente alterado.

# 2 Descrição Geral das Principais Classes / Interfaces

## 2.1 Descrição da classe AuthorizationClient

Classe responsável pelo pedido de autorização e obtenção das pastas a pesquisar.
Esta classe tem a única finalidade de obter as pastas onde poderão ser efectuadas pesquisas. Todo o processo é conseguido no construtor da classe que, através do protocolo estabelecido para a comunicação com o servidor de autenticação e autorização, *Securitas*, permite saber quais os recursos a que determinada sessão possui acesso.
Todos os erros relacionados com a comunicação com o servidor são transmitidos ao objecto que instanciou a classe *AuthorizationClient*.
A construção com sucesso de uma instância desta classe resulta num objecto com um conjunto de pastas que podem ser consultadas através da chamada ao método *getFolders*().

## 2.2 Descrição da classe ClientHandlerThread

Classe responsável por atender os pedidos dos clientes.
Esta é a classe que disponibiliza o serviço de pesquisa e que permite a leitura e o envio de dados aos clientes.

## 2.3 Descrição da classe Buscas

Ponto de entrada no serviço Buscas. Esta é a classe que contém o método *main* e que permite o arranque do servidor. É responsável pelo *parsing* dos parâmetros e pela sua correcta validação, e por instanciar a classe *Server.*

## 2.4 Descrição da classe Scanner

Classe responsável pela pesquisa dos ficheiros e/ou pastas que contenham o padrão dado pelo serviço *Securitas*.
Esta é a classe que contém o método de pesquisa e que permite procurar ficheiros e/ou pastas que correspondam ao padrão indicado.
A classe estende da classe *Thread* de modo a ser iniciada uma thread específica para a pesquisa, que é efectuada recursivamente.

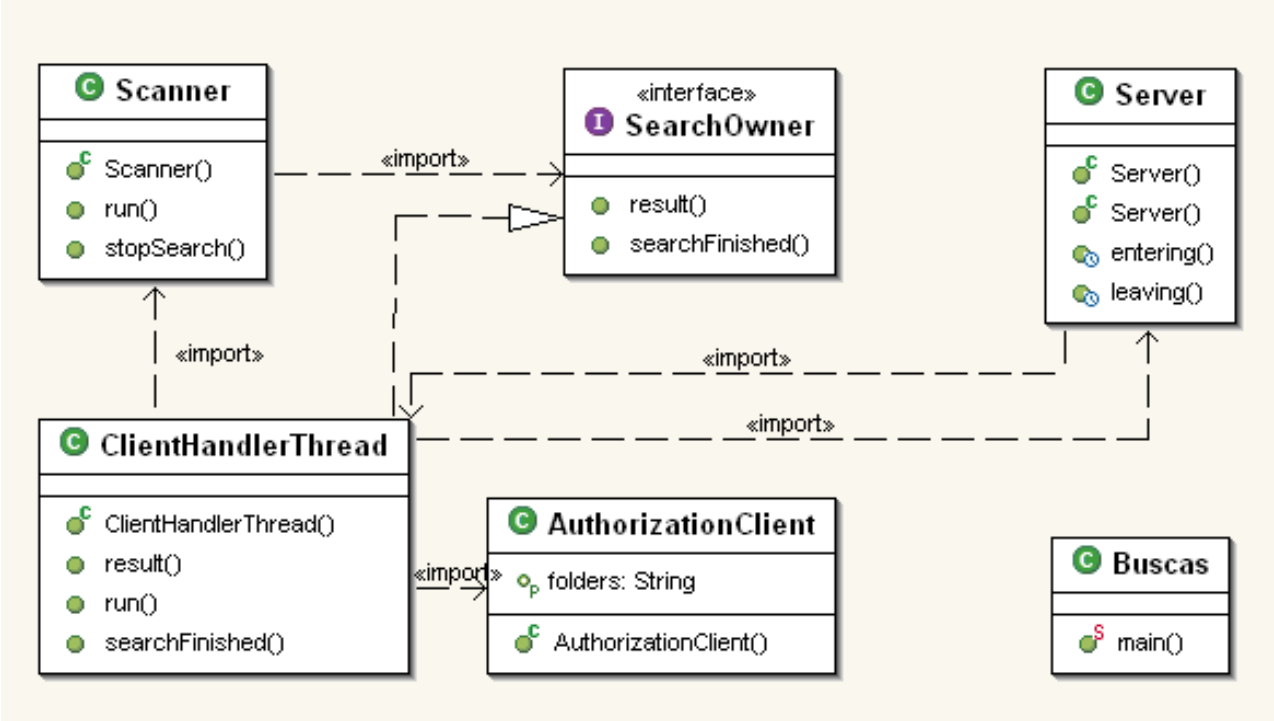
## 2.5 Descrição da classe Server

Classe que representa o servidor híbrido e que é responsável pela criação de threads que possam atender pedidos de clientes.

## 2.6 Descrição da interface SearchOwner

Interface que representa um objecto que pretende possuir um serviço de pesquisa. Permite a interoperabilidade entre a thread que efectua a pesquisa e o objecto que pediu a pesquisa.

# 3 Diagrama de Classes




- Ver ficheiro *diagram.png.*

# 4 Justificação de Opções Tomadas

Embora seja usada uma solução orientada a objectos, o problema específico não se apresenta de forma que tal abordagem possa ser levada mais seriamente, assim as classes apresentadas reflectem opções de organização do programador em vez de soluções de uma implementação orientada a objectos.

Optou-se por criar threads estendendo da classe *Thread* apenas por questão de organização. Como não acrescentamos nada à classe *Thread*, segundo o paradigma da programação orientada a objectos, deveríamos usar a interface *Runnable*, no entanto considerou-se que, no âmbito do problema, essa distinção não apresenta problemas.

# 5 Bibliografia

Oaks, Scott; Wong, Henry; Java Threads; O'Reilly 1997.

Apontamentos teóricos da disciplina de Sistemas Distribuídos e Paralelos.

"The Java Tutorial", actualização de 15 de Abril de 2005, versão de *download*.