

Licenciatura em Engenharia Informática

sistema de notificação de avisos sobre alteração nos horários dos comboios

Grupo 9

Axel Guimarães – 8180657

José Pereira

Diogo Silva

Conteúdo

[Introdução 2](#_Toc124963962)

[Descrição dos componentes implementados 3](#_Toc124963963)

[Server 3](#_Toc124963964)

[Server 3](#_Toc124963965)

# Introdução

No âmbito da avaliação prática da unidade curricular de Sistemas Distribuídos, foi proposta a elaboração de um sistema de notificação de avisos sobre alteração nos horários dos comboios. Este projeto visa a simulação de um sistema de notificação de avisos sobre a alteração nos horários dos comboios. O objetivo é o sistema notificar os intervenientes do sistema (rede ferroviária) da existência de alterações nos horários dos comboios enviando avisos mediante um evento, a incidência do mesmo e ficando à espera de receber alguma resposta.

Este projeto foi desenvolvido em linguagem de programação Java com recurso a *multithreading*, de modo a um funcionamento ordenado de recursos TCP e UDP para a comunicação entre partes.

# Descrição dos componentes implementados

Nos subtemas que se seguem, será dada uma breve descrição dos componentes desenvolvidos no projeto.

## Server

Os seguintes componentes estão incluídos no *package* Server:

### Server

Esta classe recebe as conexões e pedidos dos clientes, criando uma *thread* para cada conexão de cliente.

### Protocol

Esta classe define o protocolo de comunicação entre o cliente e o servidor. Esta classe trata de analisar o tipo de pedido que o cliente manda para o servidor, e redireciona-o para o método específico ao pedido. Como será explicado mais à frente, todos os pedidos e respostas são trocados em formato Json. A conversão destes pedidos e respostas para formato Json é feita com auxílio da biblioteca Gson.

### JsonFileHelper

Esta classe é responsável pela leitura/escrita de/em ficheiros Json que armazenam os dados relativos a passageiros e gestores locais.

### ClientHandler

Esta classe cuida de cada instância de cliente criada, e é responsável por receber e enviar todas as mensagens relativas à respetiva instância de cliente.

### ArrayListSync

Esta classe faz o papel de uma ArrayList, mas com a particularidade de todos os seus métodos serem *synchronized*, tornando-os *thread-safe*.

## userInterface

Os seguintes componentes estão incluídos no *package* userInterface:

### Client

Esta classe representa cada cliente que se pretende conectar ao servidor. Ao se iniciar uma instância de Client, é criado um socket de ligação ao servidor, com as portas 2048 para TCP e 4446 para UDP (Multicast). São iniciados dois *threads* para receber mensagens normais e mensagens *multicast* respetivamente. Além disso, é disponibilizado um método que lança uma *thread* de envio de mensagens, que pode ser chamado quando necessário.