





**Universidade do Minho**  
Escola de Engenharia

**Sistemas Operativos**  
ENGENHARIA DE TELECOMUNICAÇÕES E INFORMÁTICA

**Trabalho Prático**  
*Grupo 7*

	Miguel Fernandes Pereira A94152
	Pedro Miguel Ferreira Bártolo A92324

2022/2023

# Índice

<b>Índice</b> .....	2
<b>1. Introdução</b> .....	3
<b>2. Desenvolvimento</b> .....	4
2.1. Contextualização .....	4
2.2. Estrutura do Programa .....	4
2.3. Limitações e falhas na implementação .....	5
<b>3. Conclusão</b> .....	6

# 1. Introdução

No âmbito da unidade curricular de Sistemas Operativos foi-nos pedido a elaboração de um programa que detete casos de incêndio, fazendo a análise de lotes de imagens de satélite. Para a concretização do mesmo foi utilizada a linguagem C, recorrendo assim aos conteúdos aprendidos ao longo das aulas teóricas e práticas.

## 2. Desenvolvimento

### 2.1. Contextualização

Este projeto tem como objetivo a detecção de incêndios gerando alarmes, sendo estes enviados para o quartel de bombeiros por email.

Ao fazer a pesquisa dos pixéis deve ser definido o número máximo de incêndios que queremos detetar em cada lote de pixéis.

### 2.2. Estrutura do Programa

- **satelite.c**

O programa Satellite tem implementada a função pesquisaLote que recebe como argumentos, o nome do ficheiro (onde estão os ficheiros com os pixéis a analisar), o apontador para um array de structs do tipo coordenada (alarmes), e o número máximo de alarmes que queremos gerar em cada lote de pixéis (variável n), deste modo o array de alarmes poderá ter mais que n valores, caso encontre incêndios em diversos ficheiros *.dat*.

Para saber quantos lotes de pixéis teremos de analisar é contado o número de linhas do ficheiro *.txt*. Assim é criado um array pfd [nº linhas][2] que servirá para fazer a comunicação através de pipes entre os processos que farão a pesquisa dos incêndios, é de notar que teremos um total de (nº linhas) processos, mas simultaneamente haverá um máximo de 4.

- **pesquisa.c**

Ao executar o Pesquisa é passado na linha de comandos o path do ficheiro *.txt* em que queremos pesquisar por incêndios, e o valor máximo de alarmes a gerar em cada ficheiro *.dat*.

Na main usamos a função implementada em satellite.c para gerarmos alarmes que serão passados para a função ProcessoEmail. No final da main é libertada a memória alocada para os alarmes.

A função ProcessoEmail recebe como argumentos, duas variáveis derivadas da função main, o apontador para o array alarmes e o número máximo de alarmes que queremos gerar em cada lote de pixéis (variável n do tipo inteiro).

Inicialmente criamos um pipe(pfd) que será utilizado para comunicar entre processos. De seguida, criamos o processo filho que terá como primeiro objetivo redirecionar (através da função dup2) o stdin para a parte que dá read do pipe fd, ou seja, o pfd[0].

Como último objetivo, o processo filho vai executar (através da função execlp) o Bash Shell Script File, de nome mail.sh, que nos foi fornecido no início do projeto, de modo a poder enviar a lista de coordenadas em stdin para o email “bombeiros@protecao-civil.pt”. As coordenadas são enviadas para o stdin com um write no processo pai.

- **pesquisasrv.c**

No servidor é criado e aberto dois pipes com nome, sendo um para a comunicação via cliente-servidor e outro para a comunicação via servidor-cliente. É declarado um array de chars de 2 posições, sendo este utilizado para guardar o path e o n° máximo de alarmes recebido do cliente.

Dentro de um while(1) o servidor está constantemente a ler do pipe por onde é transmitido os dados do cliente, e caso leia alguma coisa é criado um processo filho onde é usado o execlp, tendo como argumentos, o programa Pesquisa, o path e o n° máximo de alarmes recebido do cliente. Posteriormente, no processo pai é enviado para o cliente uma mensagem a confirmar que a pesquisa foi efetuada.

Para parar o servidor é preciso que seja o utilizador a fazê-lo.

- **pesquisacli.c**

Ao executar o cliente é passado na linha de comandos o path do ficheiro *.txt* em que queremos pesquisar por incêndios, e o valor máximo de alarmes que queremos gerar em cada ficheiro *.dat*. No cliente também é criado e aberto os mesmos pipes com nome do servidor. É enviado através do fifo o path e o n° máximo de alarmes para o servidor e logo a seguir o cliente lê do outro pipe para saber se a pesquisa foi efetuada ou não, se conseguiu ler alguma coisa dá print para o stdout. Antes do cliente terminar são fechados os dois pipes com nome.

### 2.3. Limitações e falhas na implementação

O programa desenvolvido para este projeto, apesar de tudo, evidencia algumas limitações que, por consequência, limitam a eficiência do programa:

- É necessário que sejam introduzidos paths e valores reais para que os programas funcionem sem problemas.
- A visualização dos alarmes só é possível ser feita no servidor.

### 3. Conclusão

Este trabalho ajudou o grupo a consolidar, entender e a implementar os conteúdos lecionados ao longo do semestre, bem como melhorar a nossa capacidade e domínio na linguagem C.

Inicialmente começamos por abordar o projeto só tendo em conta o exemplo.txt fornecido pelos docentes, tendo depois de adaptar o código a situações de maior dimensão, alocando memória nas variáveis iniciais tendo em conta a quantidade de ficheiros *.dat* que o programa terá que procurar.

Tivemos uma maior dificuldade em implementar o servidor e o cliente, pois demoramos a compreender qual seriam as funções de cada um.

Não foram implementadas as valorizações dos objetivos 2 e 3.

Consideramos que apesar das dificuldades encontradas ao longo do projeto, obtivemos sucesso na sua conclusão.