Instituto Superior de Engenharia de Coimbra
Licenciatura de Engenharia de Informática
DEIS

Sistemas Operativos

2021/2022

Relatório de Trabalho Prático

Meta 2

Erica Wallberg n.º2017014841 Turma P11

- **Variáveis de Ambiente**

De forma a tornar persistente a inicialização das variáveis de ambiente MAXCLIENTES e MAXMEDICOS, foram adicionadas as seguintes duas linhas de código ao shell script .bashrc.

```
export MAXCLIENTES=10
export MAXMEDICOS=10
```

.bashrc é um script que corre sempre que inicializa uma sessão interativa da shell. Contém um grupo de comandos com certas preferências para obter um ambiente específico no uso da shell. Ao adicionar as duas linhas de código acima mencionadas, as variáveis MAXCLIENTES e MAXMEDICOS serão exportadas a cada sessão nova da shell, sendo possível assim mantê-las ao longo de todas as sessões. O comando export permite que os processos filhos criados pela shell tenham também acesso às variáveis de ambiente exportadas.

- **Estruturas de Dados**

As estruturas de dados principais utilizadas são as estruturas cliente e medico.

À medida que se vão conectando mais clientes e médicos, irão ser adicionadas mais posições num array dinâmico de estruturas respetivas a cada tipo de utilizador.

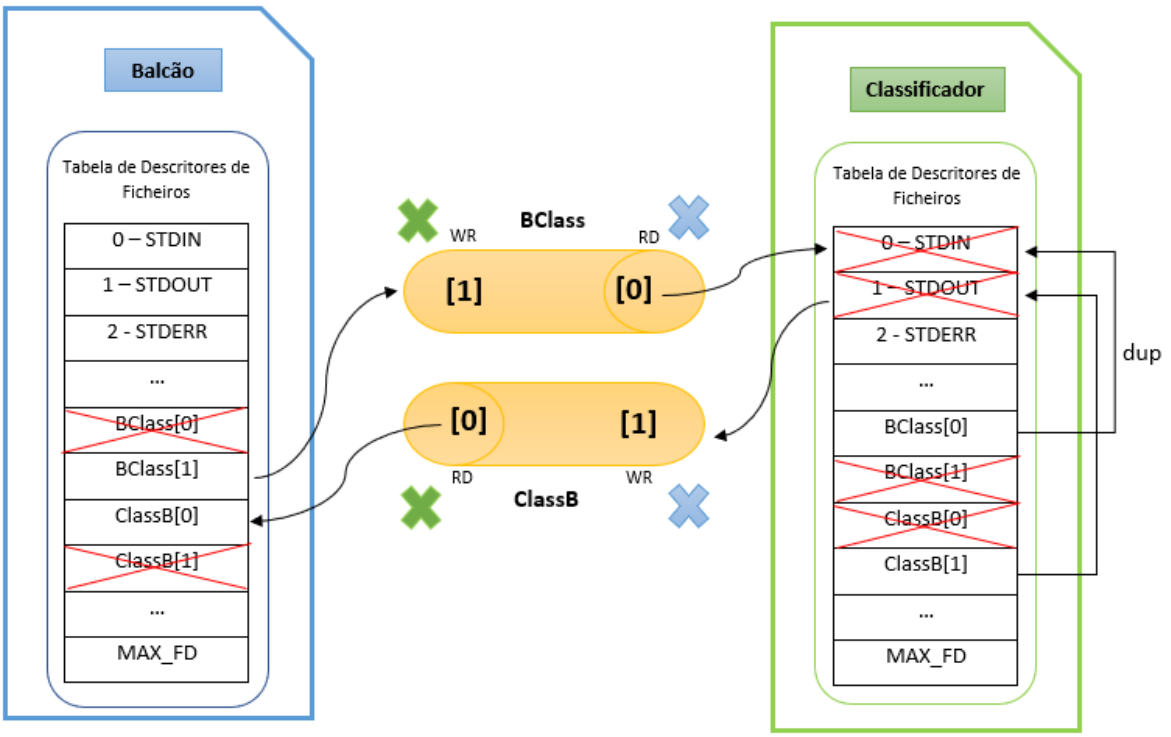
```
typedef struct cliente{
    char nomepipe[PATH_MAX];
    char nome[PATH_MAX];
    char sintomas[PATH_MAX];
} cliente;
```

Em vez de enviar o seu PID, o cliente envia logo o nome do pipe onde se encontra à escuta. A mesma estratégia foi utilizada para a estrutura medico.

```
typedef struct medico{
    char nomepipe[PATH_MAX];
    char nome[PATH_MAX];
    char especialidade[PATH_MAX];
    struct tm last_heartbeat;
}medico;
```

Para a estrutura medico, foi adicionado o campo last_heartbeat que guarda a data e hora do último registo dado pelo programa médico. Será uma estratégia que, como previsto, foi implementada na segunda meta do trabalho prático para verificar "sinais de vida" de cada médico registado no sistema.

- **Pipes Anónimos**



Este diagrama representa a lógica de pipes anónimos utilizada no trabalho prático. Nesta fase de execução, o programa cria dois descritores de ficheiro para cada pipe (BClass[0], BClass[1] e ClassB[0], ClassB[1]). De seguida, é feito um fork, duplicando-se assim, o processo (logo, a tabela de descritores de ficheiros faz também parte do processo filho).

O processo filho começa por fechar os descritores de ficheiro que não lhe interessam (a parte de escrita de BClass e a parte de leitura de ClassB). De seguida, fecha também o descritor de ficheiro correspondente ao STDIN e irá "substituir" essa posição na tabela pela parte de escrita de BClass (BClass[0]), visto que toda a informação lida pelo "stdin" deste processo será proveniente deste pipe. Por fim, a posição anterior do descritor de ficheiro BClass[0], é também fechada. Esta lógica é replicada para a posição 1 (STDOUT) da tabela de descritores de ficheiros do processo filho. De seguida, é feita uma chamada execl para que o programa do processo filho seja totalmente substituído pelo executável passado por argumento (classificador).

Por outro lado, o processo pai irá apenas de fechar a extremidade de escrita do pipe BClass (BClass[0]) e a parte de leitura do pipe ClassB (ClassB[1]). Não irá executar

chamadas dup, nem fechar as STDIN e STDOUT visto que o programa necessita destas posições para outros propósitos, ao contrário do processo filho que apenas executa o classificador.

De seguida, é perguntado ao utilizador os sintomas. A essa mensagem é adicionado um "\n" para que o programa controlador consiga ler corretamente os dados. Depois é feito um write para a posição de escrita do pipe BClass e, consequentemente, um read do pipe de "sentido contrário", ClassB.