Sistemas Operativos

Trabalho Prático

Controlo e Monitorização de Processos e Comunicação

Mestrado Integrado em Engenharia Informática
Licenciatura em Ciências da Computação
Universidade do Minho

19 de Maio de 2020

Informações gerais

- O trabalho deverá realizado em grupos de 3 elementos (máximo).
- O trabalho deverá ser submetido até às 23:59 de 7 de Junho de 2020;
- Deverá ser entregue o código fonte e um relatório de até 8 páginas (A4, 11pt) no formato PDF (excluindo eventuais capas e anexos).
- A apresentação do trabalho será posteriormente agendada para o período entre os dias 15 e 26 de Junho.
- Este trabalho tem carácter obrigatório, sendo necessário obter uma nota mínima de 10 valores (em 20 possíveis). Tem peso de 50% na avaliação final da unidade curricular de Sistemas Operativos.
- O processo de inscrição dos grupos decorrerá até ao dia 22 de Maio (será publicada informação adicional no Blackboard desta UC).

Resumo

Pretende-se que implemente um serviço de monitorização de execução e de comunicação entre processos. O serviço deverá permitir a um utilizador a submissão de sucessivas tarefas, cada uma delas sendo uma sequência de comandos encadeados por pipes anónimos. Além de iniciar a execução das tarefas, o serviço deverá ser capaz de identificar as tarefas em execução, bem como a conclusão da sua execução. Deverá terminar tarefas em execução, caso não se verifique qualquer comunicação através de pipes anónimos ao fim de um tempo especificado. O serviço deverá ainda terminar as tarefas em execução caso seja especificado um tempo máximo de execução.

A interface com o utilizador deverá contemplar duas possibilidades: uma será através de linha de comando, indicando opções apropriadas; a outra será uma interface textual interpretada (shell), e nesse caso o comando irá aceitar instruções do utilizador através do standard input.

Funcionalidade mínima

O serviço de monitorização deverá suportar, pelo menos, as seguintes funcionalidades:

- definir o tempo máximo (segundos) de inactividade de comunicação num pipe anónimo (opção `-i` n da lina de comando)

```
argus$ tempo-inactividade 10
```

- definir o tempo máximo (segundos) de execução de uma tarefa (opção `-m` n da linha de comandos)

```
argus$ tempo-execucao 20
```

- executar uma tarefa (opção `-e` "$p_1 \mid p_2 \ldots \mid p_n$" da linha de comando)

```
argus$ executar 'cut -f7 -d: /etc/passwd | uniq | wc -l'
nova tarefa #5
```

- listar tarefas em execução (opção `-l` da linha de comando)

```
argus$ listar
#1: ./a.out | teste
#3: date
#5: cut -f7 -d: /etc/passwd | uniq | wc -l
```

- terminar uma tarefa em execução (opção `-t` n)

```
argus$ terminar 1
```

- listar registo histórico de tarefas terminadas (opção `-r`)

```
argus$ historico
#2, concluida: ./a.out | ./b.out
#3, max inactividade: prog1 | prog2 | prog3
#4, max execução: prog2 | prog3
```

- apresentar ajuda à sua utilização (opção `-h`)

```
argus$ ajuda
tempo-inactividade segs
tempo-execucao segs
executar p1 | p2 ... | pn
...
```

Funcionalidade adicional

Este serviço poderá suportar, adicionalmente, a seguinte funcionalidade:

- consultar o *standard output* produzido por uma tarefa já executada (opção `-o` n da lina de comando)

```
argus$ output 3
Sat May 16 19:15:08 WEST 2020
```

A implementação da funcionalidade adicional acima descrita corresponde a 10% do valor do trabalho proposto neste enunciado. Esta componente deve permitir não só a captura e preservação num ficheiro de *log* do *standard output* produzido por cada tarefa, mas também o acesso directo a cada um desses registos (de tamanho variável). Note que o ficheiro de *log* pode crescer após a execução de cada uma das tarefas, e que pode atingir tamanhos elevados. Por esse motivo o servidor deverá manter e consultar um ficheiro adicional *log.idx*, ficheiro esse que conterá a posição onde pode ser lido cada um dos registos de *output*. Este ficheiro poderá conter informação adicional, caso necessário.

Cliente e Servidor

Deverá ser desenvolvido um cliente que ofereça uma interface com o utilizador via linha de comando que permita suportar a funcionalidade mínima descrita e ilustrada acima. O utilizador poderá agir sobre o servidor através dos argumentos passados na linha de comando do cliente, ou, no caso do cliente ser invocado sem argumentos, através de uma linguagem textual interpretada (shell). Deverá ser também desenvolvido um servidor, mantendo em memória a informação relevante para suportar a funcionalidade acima descrita.

Tanto o cliente como o servidor deverão ser escritos em C e comunicar via *pipes com nome*. Na realização deste projecto não deverão ser usadas funções da biblioteca de C para operações sobre ficheiros, salvo para impressão no *standard output*. Da mesma forma não se poderá recorrer à execução de comandos directa ou indirectamente através do interpretador de comandos (p. ex.: *bash* ou `system()`).

Note que – tal como nos exemplos apresentados acima – poderão existir múltiplas tarefas em execução num dado momento.