Universidade do Minho

Licenciatura Engenharia Informática

Sistemas Operativos (SO) – Trabalho Prático

Realizado por:

Miguel Tomás Antunes Pinto, a100815

Pedro Miguel Costa Azevedo, a100557

Samuel Macieira Ferreira, a100654

Conteúdo

Introdução

Este projeto consistiu na criação de um serviço de execução de tarefas, onde um cliente tem a capacidade de enviar tarefas para um servidor e executar. Além da execução de tarefas, o servidor permite consultar tarefas em andamento ou já executadas, visualizar os resultados de cada tarefa e encerrar manualmente as tarefas.

Inicialmente, o maior desafio foi encontrar uma maneira de metermos a comunicar o servidor com o cliente e de como iriamos definir diferentes estados. Mais tarde também encontramos certas dificuldades em obter os “status” de uma tarefa, mas com trabalho conseguimos obter resultados.

A comunicação entre o cliente e o servidor, ocorre através de dois pipes com nome: um para o envio de comandos do cliente para o servidor e outro para o envio do resultado dos comandos do servidor de volta para o cliente.

Funcionalidades disponíveis no servidor

Execução de programas do utilizador

O sistema consiste num cliente e num servidor para execução de programas. O cliente possui a opção "execute -u" para executar programas e comunica com o servidor para registrar o estado da execução.

Ao executar um programa, o cliente informa ao servidor o PID do processo, o nome do programa e o “timestamp” antes do início da execução. O cliente avisa ao utilizador do PID do programa que será executado e em seguida inicia a execução do programa.

Após a conclusão do programa, o cliente informa ao servidor o PID do processo que terminou e o “timestamp” após a execução estar concluída. O cliente também avisa o utilizador sobre o tempo de execução do programa em milissegundos.

Dessa forma, o cliente é responsável por executar programas, enquanto o servidor armazena informações sobre a execução e as disponibiliza para consulta posterior.

Status

O servidor possui uma lista que contêm todas as tarefas executadas até o momento, que inclui o nome do programa, o seu pid e o seu tempo de execução, e também contêm o estado da execução de cada tarefa. O estado pode ser: "Loading", "Finished".

PID.txt

A cada processo realizado é criado um ficheiro “.txt” com o nome do PID do processo. Dentro deste ficheiro vai ficar o nome do programa associado ao PID do nome de ficheiro texto e o tempo de execução final do programa, ou seja, o tempo total para o programa ser executado.

Testes

A seguir, compartilhamos alguns dos testes manuais que realizamos para verificar o funcionamento do nosso programa. Além desses testes, realizamos outros testes, mas os exemplos a seguir abrangem todas as principais funcionalidades do programa. Nas capturas de tela abaixo, a menos que indicado de outra forma, os comandos foram executados em sequência, um após o outro.

Teste de execução de programa único

Comando: execute -u <Program-1>

Este comando irá devolver o tempo de execução do devido programa e o devido PID do programa.

```
thomas@TuliCreme:~/a100815/2ano/Coisas de c/SO/projeto SO$ gcc tracer.c
thomas@TuliCreme:~/a100815/2ano/Coisas de c/SO/projeto SO$ ./a.out execute -u sleep 20
O PID do processo atual é: 11572.
O tempo preciso para o programa ser executado foi 20002 milissegundos.
```

Enquanto o comando está a ser executado em simultâneo o monitor dá-nos as “Estatísticas iniciais” que são o nome do programa em execução, o seu PID e o tempo inicial. Logo após o programa ser executado, este devolve-nos as “Estatísticas finais” que são na mesma o PID e o tempo Final.

```
thomas@TuliCreme:~/a100815/2ano/Coisas de c/SO/projeto SO$ gcc monitor.c
thomas@TuliCreme:~/a100815/2ano/Coisas de c/SO/projeto SO$ ./a.out
Estatísticas iniciais: sleep || 11572 || 1684008752412
Estatísticas finais: 11572 || 1684008772414
```

A meio da execução deste programa, se executarmos o “status” podemos ver os milissegundos correntes naquele momento, o nome do programa e o seu PID.

```
thomas@TuliCreme:~/a100815/2ano/Coisas de c/SO/projeto SO$ ./a.out status
sleep || 11572 || 9063 ms (Loading)
thomas@TuliCreme:~/a100815/2ano/Coisas de c/SO/projeto SO$ ./a.out status
sleep || 11572 || 10651 ms (Loading)
```

Uma vez que o programa esteja terminado, se executarmos o “status”, este vai devolver-nos o tempo final que o programa demorou a ser executado acompanhado de uma mensagem “(Finished)”, em conjunto com o nome do programa e o seu PID.

```
thomas@TuliCreme:~/a100815/2ano/Coisas de c/SO/projeto SO$ ./a.out status
sleep || 11855 || 20003 ms (Finished)
```

Escolhas Feitas

Decidimos criar uma lista ligada que acumula todos os processos e é composto por PID, nome do programa, tempo inicial, tempo final e pelo elemento que vem a seguir na lista ligada, conhecido como “next”. Sentíamos que ia ser uma maneira muita mais fácil de armazenar este tipo de dados numa lista e assim seria bastante mais fácil de procurar por determinados PIDs.

Optamos por adotar uma abordagem em que o arquivo de log, que registra as saídas das tarefas, seja excluído sempre que o servidor seja reiniciado. Essa escolha foi baseada em nossa própria decisão, pois tínhamos liberdade para optar por diferentes métodos. Vale ressaltar que tal decisão não foi influenciada pelas especificações do enunciado, as quais não faziam menção explícita a um método específico. Dessa maneira, a cada nova execução do servidor, um novo arquivo de log será gerado, seguindo essa abordagem.

A responsabilidade de criar os FIFOs é atribuída exclusivamente ao servidor, o qual deve ser iniciado antes de qualquer cliente. Embora também pudéssemos ter implementado a funcionalidade de criação dos FIFOs pelo cliente, optamos por centralizar essa tarefa no servidor. Essa escolha baseia-se no sentido de organização, uma vez que o servidor é projetado para operar continuamente, eliminando qualquer possibilidade de conflito na criação dos FIFOs. Além disso, essa abordagem permite uma melhor coordenação e controlo de todo o sistema, garantindo um funcionamento mais eficiente e estável.

Conclusão

Como é possível ver, o nosso trabalho não possui todas as funcionalidades pedidas no enunciado, mas todas as funcionalidades que temos funcionam perfeitamente e sentimos que o grupo tem um raciocínio muito bem conseguido para tudo o que está realizado. Mas ainda assim estamos satisfeitos com este trabalho pois permitiu-nos consolidar bastantes conhecimentos da cadeira de Sistemas Operativos.