· U  C · Sistemas Operativos
2020/2021

João António da Silva Melo | 2019216747

Miguel António Gabriel de Almeida Faria | 2019216809

Relatório

Simulador de Corridas

Introdução

Este trabalho prático foi realizado no âmbito da disciplina de Sistemas Operativos. Tem como objetivo desenvolver um sistema que simule uma corrida de carros, onde existem diversas equipas e carros que irão competir e onde ocorrem paragens na box para abastecer combustível ou reparar avarias. Com este relatório pretende-se explicar as opções tomadas ao longo do projeto, os recursos utilizados e a maneira como foram implementados. Relativamente às horas despendidas neste trabalho, cada um de nós investiu entre 50 a 60 horas, sendo a maior parte do trabalho realizado em conjunto, com divisão de tarefas.

Trabalho Prático

No início da aplicação, começamos por abrir o ficheiro de *logs* e criar os semáforos, message queue e memória partilhada, que vão ser utilizados ao longo do programa.

Posteriormente, recorremos à função *simuladorCorrida*, onde o ficheiro de configurações irá ser lido, mapeamos a memória partilhada e criamos dois processos: gestor de corrida e gestor de avarias. Além disso, é inicializado o unnamed pipe e a captação de sinais (que até então estavam a ser ignorados para garantir a boa inicialização das estruturas).

A função *gestorCorrida* começa por criar *unnamedPipes* que irão ser utilizados na comunicação entre os diversos processos/threads. Optámos por criar *pipes* para a comunicação entre o gestor de corrida e os carros, de modo a permitir que o gestor de corrida saiba quando o estado do carro é alterado. Dado que a corrida poderá apenas começar quando o número de equipas for o existente no ficheiro de configurações, todos os processos gestor de equipa necessários para a sua execução são previamente criados. Após "abrir" o *namedPipe*, é chamada a função *lerComandos*, que irá permitir receber comandos, criar equipas e carros, notificar as equipas do início e fim da corrida, verificar as alterações de estado dos carros, entre outros.

A função *gestorEquipas* é responsável por "criar" as *threads* carro assim que receba a indicação de "START RACE" e coloca a funcionalidade box em execução, em que

quando um carro entra, realiza a reparação ou abastecimento e envia um sinal pela variável de condição para a thread carro, de modo que este continue a sua corrida.

A função *funcaoCarros* é a função que a *thread* executa e que vai verificar quando o carro termina ou desiste. Nesta função é também executada a função corrida que é responsável pelas físicas da corrida e por determinar quando é que o carro pode entrar nas boxes.

A função *gestorAvarias* verifica por si o início e termino da corrida. Começa a gerar avarias para os carros e a notificá-los, de acordo com o pedido. O gestor de avaria apenas verifica se o carro já terminou ou desistiu, sendo a verificação de uma avaria previa existente feita pela parte do carro.

Relativamente ao tratamento de sinais, os sinais SIGINT (^C) e SIGTSTP (^Z) estão a funcionar de acordo com o pedido, no entanto o sinal SIGUSR1 não consegue receber o comando "START RACE" de novo, de modo a iniciar a corrida novamente.

O diagrama seguinte representa esquematicamente a arquitetura do nosso programa: