1 2  9 0

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D COIMBRA

Relatório Trabalho Prático de Sistemas Operativos

Simulador de Corridas

Licenciatura em Engenharia Informática

2020/2021 – 2º Semestre

Trabalho Realizado por:

Filipe Rocha Ribeiro – 2019223576

Thomas Pereira Fresco – 2019219057

Descrição do funcionamento do Programa:

Quando iniciado o programa, é efetuada a leitura do ficheiro "input.txt" que contém os dados sobre a corrida e são criados/inicializados todos os recursos necessários. De seguida, é criado, se não existir, um ficheiro "log.txt", onde vão ser registados os vários acontecimentos da corrida (início e fim do programa, input de comandos, início da corrida e possíveis problemas na corrida).

Depois, o programa fica recetivo a comandos (através de um "echo" direcionado para o named pipe, numa consola diferente), podendo, a partir deles, adicionar novos carros e começar a simulação. Durante a corrida, os carros estão sujeitos a avarias, devendo entrar nas boxes para abastecimento ou reparação, e, a qualquer altura, o utilizador pode enviar comandos (que serão rejeitados) ou sinais (SIGINT, SIGTSTP, SIGUSR1). O envio do SIGUSR1 para o processo RaceManager pode ser feito através de um simples comando "kill" ou do ficheiro "sigusr1.sh", que é atualizado com o id desse processo sempre que uma corrida nova começa.

Uma vez acabada a corrida, são impressas as estatísticas e uma das duas situações pode acontecer: o programa acaba, terminando/libertando/removendo os recursos utilizados; a corrida recomeça, dando-se um reset aos dados da corrida anterior.

Shared Memory:

Para que cada um dos processos tenha acesso a variáveis que são alteradas por outros processos, implementou-se uma zona de memória partilhada que contém informação relevante sobre a simulação.

Named Pipe:

Através do Named Pipe são recebidos dois comandos:

- "ADDCAR TEAM: A, CAR: 20, SPEED: 20, CONSUMPTION: 0.04, RELIABILITY: 95"
- "START RACE!"

O primeiro comando permite-nos adicionar carros à simulação e o segundo permite o começo da simulação. Qualquer outro comando que não cumpra os requisitos de sintaxe será rejeitado.

Unnamed Pipes:

Cada processo "Gestor de Equipa" envia informação ao processo "Gestor de Corrida" através de um Unnamed Pipe dedicado. Os dados transmitidos são relativos ao estado do carro (Corrida, Segurança, Box, Desistência, Terminado).

Message Queue:

O "Gestor de Avarias" comunica a cada um dos carros em corrida, com base numa probabilidade definida pelo campo "Reliability", informação sobre uma avaria. Essa comunicação é feita através de uma Message Queue.

Sinais:

- **SIGINT(CRTL+C) -** Termina a corrida e o programa. Aguarda que todos os carros cruzem a meta (mesmo que não seja a sua última volta) e os carros que se encontram na box no momento da instrução terminam. Após todos os carros concluírem a corrida, imprime as estatísticas do jogo e termina/liberta/remove todos os recursos utilizados.

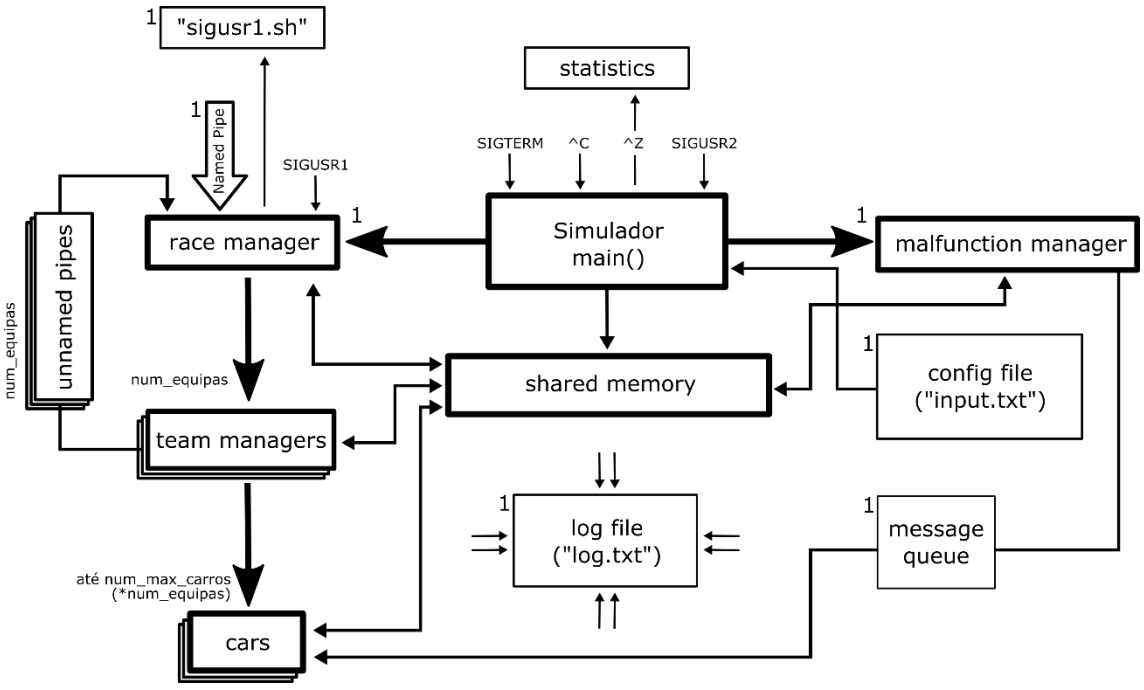
- **SIGTSTP(CRTL+Z) -** Imprime as estatísticas

- **SIGUSR1-** Interrompe uma corrida que esteja a decorrer. A corrida termina mal todos os carros cheguem à meta e apresenta a estatística final. A informação da interrupção é escrita no log. Se receber novo comando "START RACE!", a corrida reinicia.

Mecanismos de Sicronização:

- Semáforo Binário (POSIX) para escrita controlada na Shared Memory.
- Semáforo Binário (POSIX) para escrita controlada no ficheiro "log.txt".
- Mutex que salvaguarda que cada carro acede à box da sua equipa exclusivamente.
- Captação do SIGTERM como mecanismo auxiliar de terminação do programa quando a corrida acaba normalmente. O programa admite nova corrida se receber o sinal SIGUSR1.
- Captação do SIGUSR2 para a criação do processo Malfunction.
- Este último mecanismo tem o auxílio de um terceiro semáforo binário (POSIX).

Arquitetura do Programa:



Horas de Trabalho:

Filipe Ribeiro: 88h

Thomas Fresco: 100h