Sistemas Operativos 2020/2021

2º ano

Trabalho Prático

Sistema Champion UNIX

Relatório Final

Ricardo Jorge Veiga da Silva - 1998003820 - a21120616@isec.pt
Luís Manuel Antunes de Matos Viegas - 2015015646 - a21250789@isec.pt

Índice

1. Introdução

Serve o presente relatório para detalhar a implementação das funcionalidades referentes ao trabalho prático respeitando os requisitos indicados no enunciado e também para justificar as decisões tomadas quando isso se revele necessário.

É detalhada a estrutura de comunicação entre clientes e árbitro e entre árbitro e jogos dos respectivos clientes.

São também explicadas as estruturas de funcionamento de cada um dos programas, nomeadamente no que toca à implementação das threads, ao uso de mecanismos de sincronização, como sejam mutexes e variáveis condicionais, bem como o uso de sinais entre processos e threads.

2. Estrutura de comunicação

No gráfico abaixo é possível perceber a forma como os processos comunicam entre si no decorrer do campeonato, através de named pipes, no caso da comunicação entre cliente e árbitro e unnamed pipes no caso da comunicação entre árbitro e jogos.

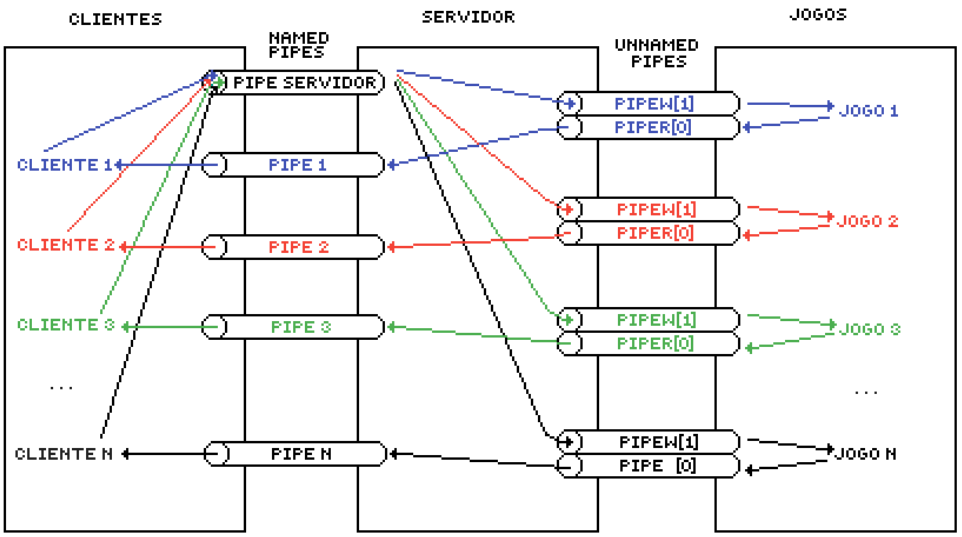

Figura 1 - Estrutura de comunicação

Inicialmente existe um pipe de login criado pelo servidor e conhecido pelos programas cliente, após a inscrição de pelo menos dois clientes e após o fim do tempo de espera este pipe é fechado e eliminado e substituído por um novo pipe também ele conhecido pelos cliente, formando a estrutura de comunicação que se observa na Figura 1.

A criação deste novo pipe tem por objectivo impedir que clientes possam estar a tentar efectuar login durante o decorrer do campeonato, qualquer tentativa de login nesta fase é derrotada à partida porque o pipe simplesmente não existe. Quando termina o campeonato voltamos ao estado inicial em que o pipe de login espera por inscrições no novo campeonato.

3. Estrutura de Threads

Neste capítulo são detalhados pormenores relativos à estruturas de threads usadas tanto no caso do servidor (o árbitro), como no caso dos clientes.

3.1. Servidor (árbitro)

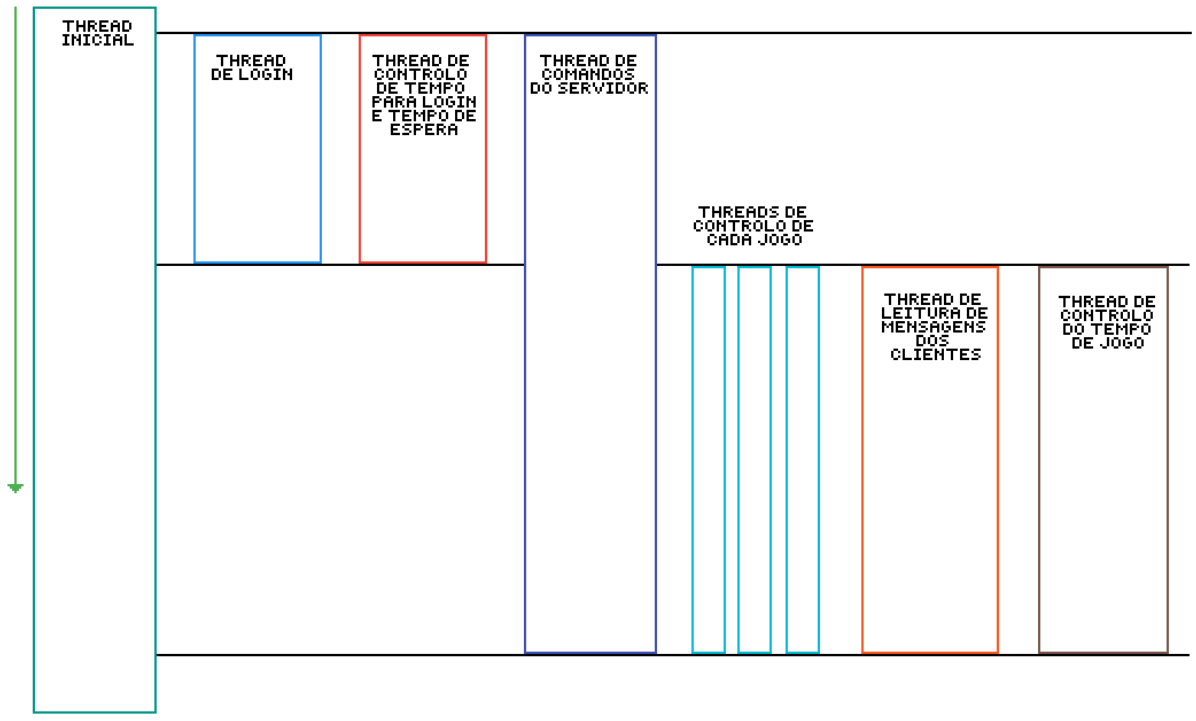

Figura 2 - Estrutura de threads do servidor

Na Figura 2 podemos perceber a estrutura geral das threads usadas no servidor bem como o seu fluxo, quando são lançadas e quando terminam, assim:

Thread inicial - Responsável por recolher e manter os dados necessários ao bom funcionamento do árbitro, todos os dados e variáveis responsáveis pelo funcionamento do programa são mantidos nesta thread e consultados pelas outras quando necessário. Mantém-se ativa durante todo o programa e é responsável por lançar e sincronizar as restantes threads. É também responsável por comunicar as pontuações aos clientes.

Thread de Login - Responsável por receber inscrições no servidor por parte dos vários clientes, decorre até haver dois clientes mais o tempo de espera, quando termina a fase de inscrição será terminada.

Thread de controlo de tempo para login e tempo de espera - Como o nome indica esta thread é responsável por controlar o processo de login, numa primeira fase encontra-se bloqueada com recurso a uma variável condicional, após a inscrição de pelo menos dois clientes passa a estar adormecida durante o tempo de espera indicado no lançamento do árbitro, findo o tempo de espera a thread é encerrada e sinaliza a thread de login para que também esta termine.

Thread de comandos do servidor - Thread de administração responsável pela interação do administrador com o árbitro através de comandos do teclado. Inicia-se (sensivelmente) ao mesmo tempo que as duas anteriores e mantém-se até ao fim do campeonato. Pode receber comandos imediatamente, excepto os que se referem ao controlo do campeonato, que, obviamente, estão disponíveis apenas durante o campeonato.
É também responsável por terminar as threads controlo de tempo quando necessário e de leitura de mensagens dos clientes embora delegue a sincronização das mesmas para a thread inicial, exceptuando quando se trata de uma thread de jogo de um cliente expulso do campeonato.

Thread de controlo dos jogos - São, na realidade, várias threads, tantas quantos os clientes inscritos no servidor, é lançada quando dá início o campeonato, é responsável por receber mensagens dos jogos e encaminhá-las para o respectivo cliente. É também responsável por recolher a pontuação final do jogo.

Thread de leitura de mensagens dos clientes - Responsável por receber as mensagens enviadas pelo cliente, no caso de comandos age em conformidade, de outra forma encaminha a mensagem para o respectivo jogo.

Thread de controlo do tempo de jogo - É responsável por terminar o campeonato quando o tempo de jogo pré-definido chega ao fim, quando isto acontece a thread sinaliza a thread de administração que trata da restante rotina de terminus das restantes threads conforme descrito acima.

3.1. Cliente

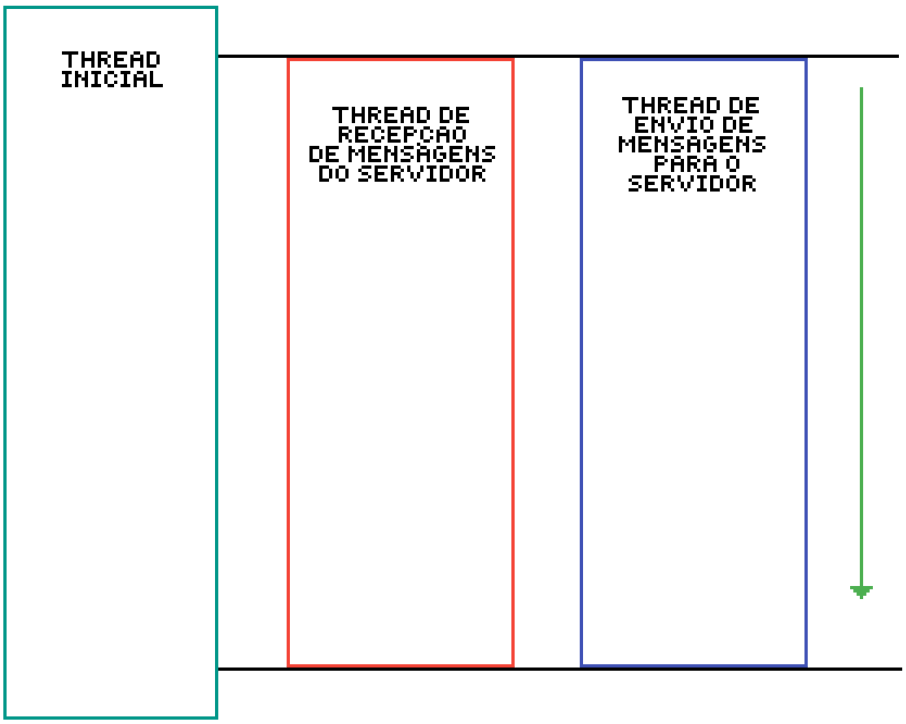

Figura 3 - Estrutura de threads do cliente

É uma estrutura mais simples do que a do servidor, o que é natural tendo em contas os requisitos de cada um dos programas, na Figura 3 podemos observar a estrutura geral de funcionamento das threads do cliente:

Thread inicial - Responsável por recolher os dados necessários ao funcionamento do cliente e por lançar e sincronizar as restantes threads.

Thread de recepção de mensagens do servidor - Responsável por receber comunicações do servidor através do seu named pipe e agir em conformidade, executar ordens ou simplesmente imprimir no ecrã as mensagens provenientes dos jogos.

Thread de envio de mensagens para o servidor - Responsável por enviar mensagens para o servidor, quer sejam comandos ou mensagens para encaminhar para os jogos.

3. Funcionalidades implementadas

Os requisitos e restrições definidos no enunciado do trabalho prático foram devidamente implementados de acordo com as especificações:

Toda a estrutura de comunicação cliente-servidor-jogo.
Todos os comandos do cliente.
Todos os comandos do servidor.
Todas as especificações relacionadas com o os tempos de espera e jogo.
Todas as especificações relacionadas com tratamento de sinais, incluindo o uso da nova API e sinais entre threads.
Todos os detalhes referentes ao fim dos campeonatos.
Toda a implementação das restrições de uso, números mínimos de jogadores, regras, pontuação, etc.

A única alteração que entendemos fazer em relação ao pedido no enunciado prende-se com o envio de sinal SIGUSR1 para os clientes em caso de fim do campeonato, do nosso ponto de vista faz pouco sentido usar este mecanismo tendo em conta que existe um named pipe aberto entre o cliente e o servidor por onde aliás se comunicam as pontuações finais e os vencedores, faz sentido portanto comunicar pelo mesmo meio o fim do campeonato, sendo uma forma de comunicação bastante mais fiável, especialmente tendo em conta que o cliente tem várias threads em funcionamento.
Outro detalhe de implementação, que entendemos não ir contra o pedido no enunciado, prende-se com o facto de os clientes serem encerrados no fim do campeonato, se o cliente tem que executar novamente login não tem interesse mantê-lo em execução, ao contrário do árbitro que, esse sim, inicia um novo campeonato findo o anterior.

5. Conclusões

Foi um trabalho bastante interessante do ponto vista do conhecimento das API's específicas dos sistemas linux bem como de toda a mecânica de aplicações com threads, sinais, mecanismos de comunicação entre processos, o seu funcionamento e boas práticas de implementação. fica ainda bastante por aprender mas nada melhor do que aplicar os conhecimentos obtidos nas aulas num projecto já razoavelmente grande para os solidificar. Tentamos seguir as melhores práticas de programação e dar o melhor uso às técnicas e API's leccionadas e esperamos ter feito um bom trabalho.