Licenciatura em Engenharia Informática

Unidade Curricular de Sistemas Operativos

Relatório Trabalho Prático

1º Ano, 2º Semestre

Dominó *online*

Carlos da Silva – 21220319

Conteúdo

Introdução

Este trabalho prático realizado ao longo do semestre tem como principal objectivo consolidar e aplicar os conhecimentos adquiridos na unidade curricular de Sistemas Operativos. Foi pretendida a elaboração de programas em linguagem C direcionada para sistemas UNIX para a construção de um sistema de jogos de dominó *online* em tempo real. A interface é em consola/modo texto e todos os jogadores estão na mesma máquina UNIX, ou seja, não se utilizaram de mecanismos de comunicação em rede – apenas mecanismos de comunicação inter-processo.

O sistema é composto por vários processos que interagem entre si segundo a lógica de cliente-servidor usando para tal mensagens, sinais e outros recursos. Um desses processos é designado de servidor e gere as regras do jogo, as peças de dominó, o estado do jogo e os jogadores e também serve de intermediário entre todos os processos envolvidos. Os restantes processos são designados de clientes, os quais têm como principal função interagir com o utilizador (jogador), servindo de ponte entre um jogador e o servidor.

O servidor armazena e gere toda a informação e controla todos os aspectos centrais do jogo. O cliente serve principalmente de interface com os jogadores e não valida as regras nem armazena informação local acerca das peças.

Para o desenvolvimento deste trabalho prático foi necessário aplicar os conhecimentos adquiridos na disciplina nomeadamente no tratamento e processamento de sinais, criação e manipulação de *pipes*, duplicação de processos (*fork*) e ainda programação C orientada para desenvolvimento de sistemas UNIX.

Estratégia e modelos seguidos

O sistema é constituído por vários programas, os quais dão origem a processos a correr numa máquina Unix. Todos os programas são lançados através da linha de comandos. Cada programa pode ser lançado a partir de uma sessão/shell diferente sem que isso afecte a sua funcionalidade, sendo sempre lançados a partir da mesma conta de utilizador da máquina. Os pormenores do lançamento e execução de cada um são os seguintes:

Servidor

Este programa corresponde ao elemento central que faz a ponte entre todos os outros, armazena e gere toda a informação com a qual o sistema lida (neste caso, um conjunto de peças de dominó, peças "na mão" de cada jogador, quais os jogadores existentes, de quem é a vez, etc.). Só deve estar a correr um processo com este programa de cada vez e se se tentar lançar um novo, o novo detecta a situação e termina. Quando lançado, passa para execução em background. Não permite estar a correr mais do que uma vez (a segunda "cópia" detecta a situação e termina logo, ficando a correr apenas a primeira. A execução de uma segunda cópia do servidor é usada para envio de ordens ao servidor apenas com carácter temporário para obter parâmetros da linha de comando e assim receber ordens (de um eventual administrador) que serão enviadas ao primeiro servidor. O servidor detecta automaticamente se já está a correr em segundo lugar ou não e, se for em segundo lugar, vai obter os argumentos de linha de comandos e envia as ordens que representam ao primeiro servidor através de *named pipes*. O segundo servidor termina imediatamente após enviar ao primeiro servidor as ordens recebidas via linha de comandos. O servidor pode ser terminado pelo envio do sinal SIGUSR2 pela linha de comandos. Quando termina, o servidor notifica os clientes para também terminarem.

As ordens suportadas via argumentos de linha de comandos para a segunda execução são:

- **show** – faz o servidor apresentar no ecrã a informação acerca do estado interno: se está a decorrer um jogo ou não, quais os jogadores logados no sistema, quais os que estão a jogar, se estiver a decorrer um jogo

- **close** – manda o servidor encerrar e os clientes são todos notificados

Cliente

Este programa permite a um utilizador juntar-se a um jogo (desde que não vá já a meio) e jogar, o que inclui toda a funcionalidade de um jogo de dominó, ou seja, consultar as peças que tem na mão, ver as peças já jogadas, obter uma nova peça e até pode desistir do jogo. O acesso de um utilizador ao sistema de jogos de dominó é livre, não necessitando de nenhuma conta previamente criada (username/password). No entanto o servidor exige a

identificação do utilizador (um nome qualquer) para efeitos de interface com o utilizador. Não existe nenhuma comunicação directa cliente-cliente. O servidor é o ponto central neste sistema e é o responsável por toda a informação e controlo de regras. O cliente pode estar a correr muitas vezes em simultâneo com ele próprio, tantas vezes quantos os jogadores. O cliente, quando se executa fica a interagir com o utilizador de forma habitual. Quando termina, avisa o servidor e mais nada de especial acontece. Os restantes processos (inclusivé outros clientes a correr) mantêm-se em execução.

Os comandos reconhecidos pelo cliente são os seguintes:

- **exit** – faz com que o programa termine e o servidor é notificado. Se o jogador estiver a meio de um jogo, aplicam-se as regras relativas ao abandono de um jogo

- **logout** – fecha a sessão do utilizador, mas o programa não termina, simplesmente revertendo à fase inicial em que lhe é pedido o nome

- **status** – apresenta no ecrã a informação acerca de se está algum jogo criado e se sim, se está a decorrer e quais os jogadores participantes

- **users** – apresenta no ecrã a informação acerca de quais são os jogadores actualmente logados e para cada um é apresentado o nome e o número de vitórias acumuladas durante a execução actual do servidor e se estão actualmente a jogar

- **new nome intervalo** – pede ao servidor para criar um novo jogo com o nome "**nome"**. O comando só é aceite pelo servidor se não estiver a decorrer actualmente nenhum jogo. O número de segundos que o servidor vai aguardar durante os quais aceita novos participantes é dado em "**intervalo"**. O número máximo de jogadores é 4. O número de peças a dar inicialmente a cada jogador é o especificado na variável de ambiente NUMPECAS. Passado o intervalo de tempo especificado, o jogo começa automaticamente se tiver pelo menos dois participantes. Se não houver pelo menos dois participantes, o jogo é cancelado

- **play** – pede ao servidor para participar no jogo e apenas tem efeito se existir nesse instante um jogo criado mas ainda não iniciado

- **quit** – desiste do jogo e apenas tem efeito se o jogador estiver a participar num jogo. As peças deste jogador que ainda não tiverem sido usadas passam para o molho (as peças que não pertencem a nenhum jogador de onde se retiram peças quando não se consegue jogar nenhuma)

Os comandos a usar no cliente, específicos ao decorrer de um jogo (e que não têm efeito se o jogador não estiver a jogar no momento), são os seguintes:

- **tiles** – mostra as peças na mão do jogador, sendo apresentadas de uma forma minimalista/modo texto e cada uma é identificada por um número que irá permitir mais tarde ao jogador identificar a peça a jogar

- **info** – mostra o número de peças no molho comum de peças, o nome dos jogadores participantes e o número de peças que cada um tem. Os jogadores são apresentados pela ordem em que jogam e é indicado o próximo a jogar

- **game** – mostra as peças já jogadas, apresentadas com um aspecto linear

- **play num left | right** – joga a peça identificada pelo número "**num",** sendo que o numero foi apresentado no comando tiles. A peça é jogada à esquerda ou à direita das que já se encontram jogadas consoante foi escrito **"left"** ou "**right"** (escreve-se ou left ou right mas não ambos e o caracter "|" também não se escreve). O servidor valida se a peça pode mesmo ser jogada, sendo testada de duas maneiras: assumindo a peça com o valor a numa ponta e b na outra – peça (a,b), o servidor verifica se tanto o valor a como o b "encaixa" na extremidade especificada (left/right). A peça (a,b) é a mesma peça que (b,a) e tanto pode ser jogada como (a,b) como (b,a). A peça é jogada e apresentada no ecrã da forma que encaixar. Se não encaixar o jogador deve escolher outra peça e se nenhuma servir tem que obter uma nova, mas se já não houver, tem que passar a vez

- **get** – obtém uma nova peça do molho e o servidor verifica se o jogador precisa mesmo de uma peça nova. Se houver uma peça na mão do jogador que possa ser jogada, o comando não tem efeito. A peça obtida é sorteada pelo servidor de entre as que estão no molho

- **pass** – prescinde da jogada e apenas pode fazer isso se não tiver nenhuma peça que possa ser jogada e não houver mais nenhuma no molho, sendo que o servidor verifica estas condições

- **help** – mostra as peças que podem ser jogadas para o caso de jogador estar com dificuldades em ver isso por si só

- **giveup** – desiste do jogo e as peças deste jogador que ainda não tenham sido jogadas passam para o molho

A estratégia base de comunicação entre o servidor e o cliente reside numa string de comando (por ex.: "exit") que é guardada numa estrutura do tipo PEDIDO e que é enviada pelo cliente para o servidor que, após receber a string de comando, realiza a algoritmia associada e envia um valor inteiro como código de comando (por ex.: -1) dentro da estrutura do tipo RESPOSTA, que o cliente interpreta e realiza então a algoritmia associada a esse código.

Juntamente com o código de resposta são também actualizados valores dentro da mesma estrutura RESPOSTA aos quais o cliente poderá aceder, sejam os nomes dos jogadores, os PIDs de cada, ou até mesmo o número de peças que ainda possuem. Em resumo, o servidor espera sempre o mesmo tipo de estrutura assim como o cliente também espera o mesmo tipo, PEDIDO para o primeiro e RESPOSTA para o segundo.

Ficheiros de programa

- **Servidor.c** – ficheiro que contêm o código relativo ao servidor

- **Servidor2.c** – ficheiro que contêm o código relativo ao segundo servidor

- **Cliente.c** – ficheiro que contêm o código relativo ao cliente

- **Util.h** – ficheiro de cabeçalho que contem a declaração das funções e estruturas utilizadas nos programas

Conclusões

A resolução deste trabalho prático contribui para a melhor compreensão da matéria lecionada nas aulas teóricas e práticas, percebendo como funciona a comunicação de programas na mesma máquina e como funciona o sistema de pedidos e respostas.

Durante a realização do mesmo apareceram algumas dificuldades nomeadamente ao nível da comunicação entre programas e como poderia ser implementada essa comunicação. Existem algumas limitações ao nível da transição de blocos de código no servidor, por exemplo, um jogo só pode começar depois de entrar um ultimo jogador após finalizado o *timeout*, pois não foi possível realizar este passo automaticamente. Da mesma forma, a desistência de um jogo implica a saída do cliente e fecho do respectivo programa, não sendo possível o regresso ao jogo e acumulação de vitórias, mesmo existindo uma tentativa de implementação desta função.