Sistemas Operativos 1

Meta 2

2020-2021

TheForgotten
JOSEALM3IDA

Licenciatura de Engenharia Informática
8 de dezembro de 2020

Índice

1 Mudanças realizadas inicialmente

Após analisar o documento de feedback geral fornecido, fizemos algumas alterações em todas as partes do trabalho. Estas são alterações que na verdade dizem respeito ainda à meta 1.

Alterações no árbitro:

- Melhoria na validação de inputs.

Alterações no cliente:

- Melhoria na validação de inputs.

Alterações no jogo proposto (unscrambler):

- Adição duma pequena descrição sobre o jogo e um pequeno tutorial.

2 Mudanças realizadas no makefile

As alterações feitas no makefile foram bastante simples e na verdade pouco alteram o seu modo de funcionamento. Estas alterações consistem maioritariamente em organizar melhor o código do mesmo e separar mais claramente as etapas e as dependências.

Para além disto adicionamos também uma rule nova chamada *clean-pipes*. Esta revelou-se bastante útil nesta fase de manipulação de pipes pois agilizou o processo da eliminação dos mesmos em caso de haver falha no programa.

Deste modo, o makefile apresenta as seguintes rules:

Rule	Dependencies	O que faz
all	arbitro cliente jogos	Compila todos os programas associados ao trabalho
debug	arbitro-debug cliente-debug	Compila todos os programas (menos os jogos) associados ao trabalho para debugging
cliente	cliente.o utils.o	Compila o programa cliente
cliente.o	cliente.c utils.h	Compila o objeto cliente
cliente-debug	cliente.c utils-debug	Compila o programa cliente para debugging
arbitro	arbitro.o utils.o	Compila o programa arbitro
arbitro.o	arbitro.c arbitro.h utils.h	Compila o objeto arbitro
arbitro-debug	arbitro.c arbitro.h utils-debug	Compila o programa arbitro para debugging
utils.o	utils.c utils.h	Compila o objeto utils
utils-debug	utils.c utils.h	Compila o objeto utils para debugging
jogos	$(GAMES)	Responsável por pedir para compilar todos os jogos do subdiretório Games
g_%	g_%.o	Compila executáveis que comecem por "g_". Usado para compilar os executáveis dos jogos
g_%.o	g_%.c	Compila objetos que comecem por "g_". Usado para compilar os objetos dos jogos
clean	clean-obj clean-exe	Limpa todos os ficheiros gerados pelo makefile
clean-obj	N/A	Limpa todos os ficheiros .o gerados pelo makefile
clean-exe	N/A	Limpa todos os executáveis gerados pelo makefile
clean-pipes	N/A	Limpa todos os pipes que possam ter sido deixados por um erro de execução

Fora as alterações mencionadas o makefile mantem-se exatamente igual.

3 Alterações no jogo proposto (unscrambler)

No jogo que foi proposto foram realizadas algumas alterações de modo a implementar o que foi requisitado.

- Realização do mecanismo de término aquando a receção do sinal SIGUSR1;
- Devolução do score / resultado através do *exit status*.

4 Lógica de comunicação cliente-arbitro através de named pipes

A proposta para a implementação da comunicação usando named pipes foi a seguinte:

1. Cliente manda o seu nome e PID para um pipe predefinido do árbitro e abre 2 pipes: um para escrita (por parte do server, ou seja, leitura para o cliente) e um para leitura (por parte do server, ou seja, escrita para o cliente);
2. Árbitro verifica se o nome é único e envia confirmação de que está tudo ok;
3. Se todas as condições se verificarem e tudo tiver corrido bem o cliente fica a comunicar com o pipe e vice versa.

A ideia é que para cada cliente existam 2 pipes, de modo a evitar leituras / escritas indevidas. Todas as flags, conceitos e funções em comum no árbitro e cliente ficam presentes num novo conjunto de ficheiros chamados utils (utils.c e utils.h).

Por enquanto os pipes estão implementados num sistema de select, mas o plano é que no futuro estes sejam implementados num sistema de threads, ou seja, para cada novo cliente cria-se uma nova thread.

5 Comandos suportados pelo cliente

Sempre que o árbitro recebe do cliente uma string que comece com o caracter '#' este vai verificar se é comando válido. Se sim, envia a informação desejada ao cliente. Caso contrário este envia uma mensagem a dizer que o comando é inválido.

Comando	**Uso**	**O que faz**
mygame	#mygame	Imprime o nome do jogo associado ao jogador
quit	#quit	Fecha o cliente avisando o árbitro

A verificação do comando foi implementada de modo a que a capitalização da palavra do comando não importe, ou seja, quer o cliente envie *"#mygame"* ou *"#mYgAmE"* o árbitro vai sempre ser capaz de interpretar o comando corretamente.

Aquando o uso do comando quit, o cliente é forçado a avisar o árbitro que desistiu de modo a que este se possa libertar de informações relacionadas com o desistente (como fechar pipes e remover da lista ligada).

Para além disto, mesmo que o cliente tente ser fechado desordeiramente, ou seja, por Ctrl+C, este continuará a fechar ordeiramente pois está preparado para lidar com sinais do tipo SIGINT.

6 Comandos de administração do árbitro

Para esta meta apenas foram implementados 4 comandos do lado do árbitro: *'players'*, *'games'*, *'k'* e *'exit'*.

Comando	**Uso**	**O que faz**
players	players	Lista todos os jogadores ligados de momento ao árbitro
games	games	Lista todos os jogos disponíveis
k	k<nome de jogador>	Remove o jogador do campeonato (exemplo: krui)
exit	exit	Fecha o campeonato e o árbitro avisando todos os clientes do sucedido

Tal como no cliente, os comandos no árbitro não dependem de capitalização, ou seja, se o administrador inserir o comando *"players"* ou o comando *"pLaYeRs"* o árbitro vai ser capaz de interpretar ambos corretamente.

Aquando o uso do comando exit, o árbitro é forçado a avisar todos os clientes de que vai fechar de modo a que estes possam agir em conformidade.

Para além disto, mesmo que o árbitro tente ser fechado desordeiramente, ou seja, por Ctrl+C, este continuará a fechar ordeiramente pois está preparado para lidar com sinais do tipo SIGINT.