Licenciatura em Engenharia Informática

Curso Engenharia Informática

Ramo de Desenvolvimento de Aplicações

Unidade Curricular de Sistemas Operativos 2

Ano Letivo de 2023/2024

Bolsa de Valores Online

Meta 2

Diogo Rafael Abrantes Oliveira

2021146037

Lara Filipa da Silva Bizarro

2021130066

Coimbra, 18 de abril de 2024

Índice

1. Introdução

O trabalho prático consiste na implementação de vários programas, com o objetivo de simular uma Bolsa de Valores Online, com o uso de uma Bolsa, Board e Cliente.

2. Arquitetura Geral

A aplicação da Bolsa é responsável por regular toda a gestão de dados e de operações, este programa interage com um "administrador da bolsa".

Quando iniciar o Cliente pela primeira vez irá pedir ao utilizador para realizar o seu login, de seguida irá ter acesso aos comandos que poderá realizar ao longo do uso da Bolsa.

A Board será simplesmente para apresentar ao Cliente as mudanças que irão acontecer, em relação ao número de Empresas, ao longo que estas sofrem alterações.

A BoardUI é um programa para apresentar o utilizador um gráfico de barras das 10 empresas, contém também um menu de personalização, onde o utilizador pode definir o mínimo e máximo valor do gráfico e a quantidade de empresas que pretende visualizar no gráfico de barras, no menu existe a informação do trabalho realizado.

3. Mecanismo de Comunicação

Nos Mecanismos de Comunicação foi usado **NamedPipes** para realizar a comunicação entre o programa do Cliente e o programa da Bolsa, **Shared Memory** para efetuar a comunicação do programa da Bolsa e o programa da Board e da BoardGUI.

4. Mecanismos de Sincronização

Os mecanismos de Sincronização usados foram Mutexes no Cliente para permitir uma saída ordenada, na Bolsa na inicialização da Memória Partilhada, no envio dos dados para a Memória Partilhada, a enviar mensagens aos clientes a enviar mensagens aos clientes e a permitir uma saída ordenada, na Board para na inicialização da Memória Partilhada, para receber mensagens da Bolsa. Foram usados Eventos para ler mensagens do cliente, para fazer parte do namedpipe, no Cliente foi usado para enviar, receber mensagens validar comandos e para permitir uma saída ordenada, na Board para inicializar a Memória Partilhada, para receber mensagens.

Por fim, usamos a estrutura OVERLAPPED tanto no lado do servidor (Bolsa) quanto no lado do Cliente. No Cliente, OVERLAPPED, em conjunto com a função WaitForMultipleObjects, permite que o cliente realize várias tarefas simultaneamente sem precisar esperar pela resposta do servidor para cada operação de E/S. Isso é especialmente útil em aplicações que precisam gerenciar múltiplas operações de rede ou de arquivo ao mesmo tempo, melhorando a eficiência e a responsividade do sistema.

5. Estruturas

```
typedef struct _Empresas {
    TCHAR nome[50];
    unsigned int numAcoes;
    float precoAcao;
}Empresas;
```

A Estrutura das Empresas, serve para guardar o nome, o número de ações e preço de cada ação.

```
typedef struct _SharedMemory {
    Empresas empresas[NEMPRESAS];
    TCHAR UltimaTransacao[100];
    int posEmpresa;
} SharedMemory;
```

A Estrutura do SharedMemory, serve para um array de ponteiros da Empresas, um Tchar para guardar a ultima Transação realizada na bolsa e um inteiro para saber a posição da Empresa.

```
typedef struct _DadosSM {
    HANDLE hMapFile;
    HANDLE shMutex;
    HANDLE shEvento;
    SharedMemory* sMemory;
}DadosSM;
```

A Estrutura dos DadosSM, contem um Handle para mapear o Ficheiro, um Handle de um Mutex para garantir que apenas uma thread aceda à memória partilha, um Handle de um Evento, que é usado quando é enviado informação para Shared Memory.

```
typedef struct _CarteiraCliente {
    TCHAR nomeEmpresa[50];
    int nAcoes;
}CarteiraCliente;
```

A Estrutura da CarteiraCliente, serve para guardar o nome da empresa e a quantidade de ações que o cliente contém.

```
typedef struct _Cliente {
    TCHAR nome[50];
    TCHAR password[50];
    float saldo;
    CarteiraCliente carteira[5];
    struct _Cliente* prox;
} Cliente;
```

A Estrutura da Cliente, serve para guardar o nome a password do cliente, o seu saldo, a carteira representa que o cliente pode conter no máximo 5 empresas e o prox tem como objetivo ser um ponteiro para o próximo cliente da lista.

```
typedef struct _Dados {
    TCHAR mensagem[2000];
    BOOL validacaoLogin;
    // divisão de comandos
    TCHAR parametro1[50];
    TCHAR parametro2[50];
    TCHAR nomeCliente[50];
    int opcao;
    float saldo;
} Dados;
```

A Estrutura Dados, tem uma mensagem para enviar mensagens, para verificar a validação do Login e a divisão de comandos.

```
typedef struct _NamedPipes {
    HANDLE hInstancia;
    BOOL ativo;
    Dados aux;
    OVERLAPPED overlap;
}NamedPipes;
```

A estrutura de NamedPipes, contem um Handle para ser usado na criação do namedpipe e no envio de mensagens, um Bool para verificar se o namedpipe está a ser usado, um aux da estrutura dados que está associado a instância do namedpipe e a estrutura do Overlapped.

```
typedef struct {
    NamedPipes hPipes[NCLIENTES];
    HANDLE hEvents[NCLIENTES];
    HANDLE hMutex;
    HANDLE hThread[NCLIENTES + 1];
    int terminar;
    DadosSM cdados;
    NamedPipes* naux;
    Cliente* lista;
    Empresas* lempresa;
    SharedMemory shm;
} ThreadDados;
```

A estrutura de ThreadDados, contem hPipes da estrutura previamente referida para armazenar as várias instâncias do pipe, um evento, um mutex e uma thread para ser usado como mecanismo de sincronização, um inteiro que funciona como trinco e resto de variáveis referentes a outras estruturas.

6. Decisões Tomadas

As decisões tomadas em relação à variância do preço das ações ao longo das vendas e das compras, decidimos se os números de ações forem maior que 100 unidades é realizado uma percentagem randomizada entre 0.4 e 0.6, se não for superior a 100 unidades a percentagem é randomizada entre 0.05 e 0.39, de seguida se o preço aumentar, é aumentado com a percentagem calculada anteriormente, mas se o preço diminuir, é diminuído com a percentagem calculada anteriormente.

7. Requisitos Implementados

ID	Funcionalidade	Estado
1	Memória partilhada entre a board e a bolsa	Totalmente implementado
2	Número de empresas a mostrar na board é recebido pela linha de comandos	Parcialmente implementado
3	Empresas mostradas na board são atualizadas sempre que ocorre uma mudança de valor de ações ou são adicionadas novas empresas	Totalmente implementado
4	Board fecha automaticamente quando a bolsa fecha	Totalmente implementado
5	Board apresenta a última transação realizada	Totalmente implementado
6	Comando addc (bolsa)	Totalmente implementado
7	Comando listc (bolsa)	Totalmente implementado
8	Comando stock (bolsa)	Totalmente implementado
9	Comando users (bolsa)	Totalmente implementado

10	Comando pause (bolsa)	Totalmente implementado
11	Comando close (bolsa)	Totalmente implementado
12	Comando de leitura de ficheiro de utilizadores(bolsa)	Totalmente implementado
13	Utilizadores são lidos de um ficheiro quando o programa bolsa é executado	Totalmente implementado
14	Named Pipes entre a bolsa e os clientes	Totalmente implementado
15	Comando login (cliente)	Totalmente implementado
16	Comando listc (cliente)	Totalmente implementado, as vezes ocorre um bug.
17	Comando buy (cliente)	Totalmente implementado
18	Comando sell (cliente)	Totalmente implementado
19	Comando balance(cliente)	Totalmente implementado
20	Comando exit(cliente)	Totalmente implementado
21	Carteira de ações para cada cliente	Totalmente implementado
22	Clientes podem apenas ter ações em 5 empresas diferentes	Totalmente implementado
23	Compras e vendas são pedidas pelos clientes e processadas na bolsa	Totalmente implementado
24	Valores de ações são alterados cada vez que uma compra/venda ocorre	Totalmente implementado
25	Implementação da BoardUI	Parte Gráfica feita, mas sem sharedmemory
26	Sempre que os valores das ações são alterados todos os clientes são notificados	Totalmente implementado

8. Conclusão

Ao longo deste trabalho, deparámos-mos e fomos resolvendo vários desafios que não esperávamos ter, tratando – se de uma excelente oportunidade para consolidação de matéria das aulas teóricas e práticas de Sistemas Operativos 2.