Licenciatura em Engenharia Informática

Instituto Superior de Engenharia de Coimbra

# Sistemas Operativos 2 2023/2024

# Bolsa de Valores Online

Rafael Filipe Martins Alves 2014013189

Ruben Filipe Estima de Almeida 2017017386

# Índice

# Introdução

Este projeto, realizado na cadeira de Sistemas Operativos II, no ano letivo 2023/2024 teve como principal objetivo a implementação de uma Bolsa de Valores Online.

Neste relatório explicamos de forma mais detalhada as estratégias de implementação para cada abordagem e a respetiva explicação. Também dispomos de um pequeno manual de instruções, em que é ilustrado os comandos que se poderá efetuar, tanto no Cliente como no Bolsa (servidor), de modo a alterar o comportamento da Bolsa.

# Mecanismos de comunicação

A relação entre o programa bolsa (servidor) e o programa cliente é a de cliente-servidor e desta forma a comunicação entre Servidor e Cliente é através de *Named Pipes*.

No programa bolsa (servidor) existe uma *thread* chamada *Input* que fica responsável por receber os comandos do administrador do bolsa.

```
DWORD WINAPI Input(LPVOID data)
```

Existe ainda uma segunda *thread* chamada *AtendeCliente* que fica responsável por atender o cliente.

```
DWORD WINAPI AtendeCliente(LPVOID data)
```

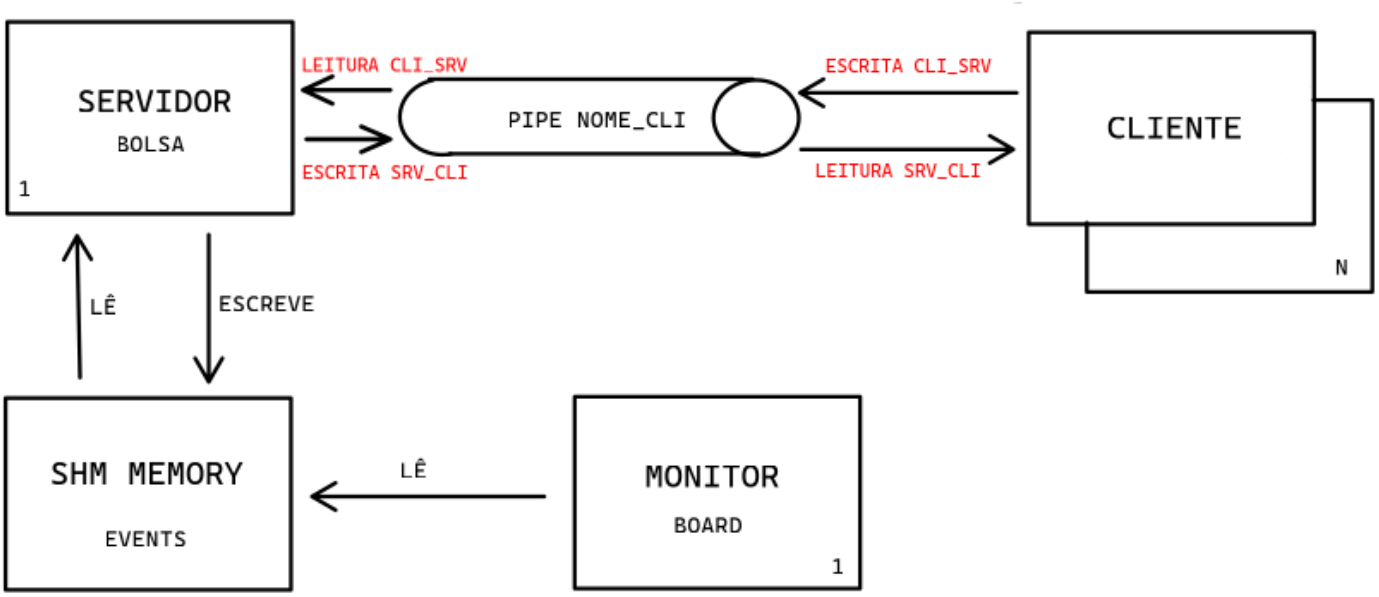
A comunicação entre o programa bolsa (servidor) e o programa monitor fez-se exclusivamente através de Memória Partilhada que se encontra no *main* do bolsa.

```
CreateFileMapping(INVALID_HANDLE_VALUE, NULL, PAGE_READWRITE, 0, MAX_EMPRESAS * sizeof(SDATA), NOME_MEMORIA_PARTILHADA);
```

No monitor existe uma *thread* chamada *OperacaoRecente* que espera por uma operação recente (compra de ações, venda de ações, e as empresas registadas).

```
DWORD WINAPI OperacaoRecente(LPVOID data)
```

De seguida, ilustramos uma imagem relativa a esta explicação:




## Defines e Estruturas

#define NOME_PIPE _T("\\\\.\\pipe\\pipe_bolsa") – Nome do *pipe* usado no Servidor-Cliente;

#define NOME_MEMORIA_PARTILHADA _T("SharedMemory") – Nome da memória partilhada usada para comunicação entre o Servidor e o Monitor;

#define NOME_MUT_UM_SERVIDOR _T("Um_Servidor") – Nome do *mutex* usado para a criação de apenas um servidor;
#define NOME_EVENTO_SERVIDOR_MONITOR _T("EventoServidorMonitor") – Nome do evento;
#define NOME_EVENTO_ALTERACAO_EMPRESAS _T("EventAlteracaoEmpresas")
#define NOME_EVENTO_ESCRITA_COMPLETA _T("EventEscritaCompleta")

#define MAX_UTILIZADORES 20 – Número máximo de utilizadores registados no programa bolsa;

#define MAX_EMPRESAS 30 – Número máximo de empresas registadas no programa bolsa;
#define MAX_SERVIDORES 1 -

#define MAX_CLIENTES 5 – Número máximo de Clientes (processos) que o servidor pode atender

```
typedef struct {
    TCHAR username[TAM];
    TCHAR password[TAM];
    float saldo;
    DWORD ligado;
    Carteira carteira[5];
    DWORD nCarteira;
    DWORD id_cliente;
} Utilizador;
```

Estrutura Utilizador responsável por armazenar as informações relativas ao utilizador e uma carteira de ações do mesmo.

```
typedef struct {
    TCHAR empresa[TAM];
    DWORD nAcoes;
} Carteira;
```

Estrutura Carteira responsável por armazenar o número de empresas e o número de ações compradas por cada cliente.

```
typedef struct {
    DWORD numUtilizadores;
    Utilizador utilizadores[MAX_UTILIZADORES];
} Cliente;
```

Estrutura Cliente responsável por armazenar o número de utilizadores e um *array* de utilizadores.

```
typedef struct {
    TCHAR nome[TAM];
    DWORD AcoesDisponiveis;
    float preco;
} Empresa;
```

Estrutura Empresa responsável por armazenar as informações das empresas.

```
typedef struct {
    DWORD numEmpresas;
    Empresa empresas[MAX_EMPRESAS];
} Mercado;
```

Estrutura Mercado responsável por armazenar o número de empresas registadas e um *array* de empresas.

```
typedef struct {
    TCHAR nome[TAM];
    DWORD nAcoes;
    float preco;
} Operacao;
```

Estrutura Operação responsável por guardar a informação sobre a transação mais recente.

```
typedef struct {
    Operacao recente;
    Mercado mercado;
    BOOL continua;
} SDATA;
```

Estrutura SDATA responsável por armazenar a informação para a memória partilhada.

```
typedef struct {
    Cliente* clientes;
    Mercado* mercado;
    BOOL continua;
    HANDLE hEvEmpresas;
    SDATA* shm;
    BOOL pause;          //Variável para verificar pause
} TDATA;

typedef struct {
    HANDLE hPipe[MAX_CLIENTES];
    HANDLE hMutex;
    HANDLE hEvOperacoes;
    DWORD id;
    TDATA* Share;
} TDATA_EXTRA;
```

Estrutura TDATA responsável por guardar a informação da *thread Input* (responsável por receber os comandos do administrador do bolsa). Estrutura TDATA_EXTRA responsável por armazenar a informação da *thread AtendeCliente*.

# Manual de Utilização

## -Servidor

Comando *users* que mostra os utilizadores lidos do ficheiro de texto (utilizadores) na execução do servidor.

```
C:\Users\rafaa\Desktop\Nova pasta (2)\project1\x64\Debug>Servidor.exe utilizadores.txt
                        ----------------------
                         BEM-VINDO AO SERVIDOR
                        ----------------------

[SERVIDOR] Chave foi Aberta!
Comando: users
Cliente: ruben | Saldo: 150.00€ | Offline
Cliente: rafael | Saldo: 200.00€ | Offline
Cliente: nunes | Saldo: 250.00€ | Offline
Cliente: duraes | Saldo: 50.00€ | Offline
Cliente: sa | Saldo: 120.00€ | Offline
Comando:
```

Comando *emp* (comando escolhido por nós, uma vez que não foi especificado no enunciado) que lê o ficheiro de texto empresas.

```
Comando: emp
Empresas inseridas com sucesso! Total de empresas:3
Empresa: edp | Ações Disponíveis: 100 | Preço: 0.50€
Empresa: tap | Ações Disponíveis: 200 | Preço: 0.25€
Empresa: ana | Ações Disponíveis: 150 | Preço: 0.60€
Comando:
```

Comando *addc*: acrescentar uma empresa.

```
Comando: addc novamepresa 300 5
Comando:
```

Comando *listc*: listar todas as empresas.

```
Comando: listc
Nome: edp | Numero de Ações: 100 | Preço: 0.50€
Nome: tap | Numero de Ações: 200 | Preço: 0.25€
Nome: ana | Numero de Ações: 150 | Preço: 0.60€
Nome: novamepresa | Numero de Ações: 300 | Preço: 5.00€
Comando:
```

Comano *pause*: pausar as operações de compra e venda.

```
pause 20
Operações suspensas por 20 segundos...
Li do cliente [1] a msg: buy tap 4
Tempo de pausa expirado! Operações retomadas!
Comando:
```

```
Comando: buy tap 4
Escrevi no pipe a msg [buy tap 4]
[MSG BOLSA] Operações suspensas!
Comando:
```

Comando *stock*: redefinir custo das ações de uma empresa.

```
Comando: listc
Nome: edp, Numero de Ações: 50, Preço: 0.50€
Nome: tap, Numero de Ações: 200, Preço: 0.25€
Nome: ana, Numero de Ações: 100, Preço: 0.60€
Comando: stock ana 100 0.75
Comando: listc
Nome: edp, Numero de Ações: 50, Preço: 0.50€
Nome: tap, Numero de Ações: 200, Preço: 0.25€
Nome: ana, Numero de Ações: 100, Preço: 100.00€
Comando:
```

## -Cliente

Comando *login*: autenticar um utilizador.

```
                        --------------------
                         BEM-VINDO CLIENTE
                        --------------------

[CLIENTE] Conectei-me ao pipe
Comando: login ruben c1sco
Escrevi no pipe a msg [login ruben c1sco]
[MSG BOLSA] Login com sucesso!
Comando:
```

Comando *listc*: listar todas as empresas.

```
Comando: listc
Escrevi no pipe a msg [listc]
[MSG BOLSA] Nome: edp | Ações Disponíveis: 100 | Preço: 0.50
Nome: tap | Ações Disponíveis: 200 | Preço: 0.25
Nome: ana | Ações Disponíveis: 150 | Preço: 0.60
Nome: novamepresa | Ações Disponíveis: 300 | Preço: 5.00

Comando:
```

Comando *buy*: comprar ações.

```
Comando: buy edp 10
Escrevi no pipe a msg [buy edp 10]
[MSG BOLSA] Compra realizada com sucesso!
Comando:
```

Comando sell: vender ações.

```
Comando: sell ana 10
Escrevi no pipe a msg [sell ana 10]
[MSG BOLSA] Venda realizada com sucesso!
Comando:
```

## -Monitor

```
C:\Users\rafaa\Desktop\Nova pasta (2)\project1\x64\Debug>monitor.exe
                        ----------------------
                         BEM-VINDO AO MONITOR
                        ----------------------


Empresa Adicionada:
Empresas Da Bolsa:
Nome: edp | Açoes: 100 | Preço: 0.50€
Nome: tap | Açoes: 200 | Preço: 0.25€
Nome: ana | Açoes: 150 | Preço: 0.60€

Empresa Adicionada:
Empresas Da Bolsa:
Nome: edp | Açoes: 100 | Preço: 0.50€
Nome: tap | Açoes: 200 | Preço: 0.25€
Nome: ana | Açoes: 150 | Preço: 0.60€
Nome: novamepresa | Açoes: 300 | Preço: 5.00€
```

# Requisitos Implementados

| Recurso | Implementado | Não Implementado |
|---|---|---|
| Comunicação Cliente-Servidor | ✓ | |
| Comunicação Servidor-Monitor | ✓ | |
| Comandos servidor** | ✓ | |
| Comandos cliente | ✓ | |
| Leitura do ficheiro utilizadores | ✓ | |
| Leitura do ficheiro empresas | ✓ | |
| ** Comando exit | | ✗ |
| GUI | | ✗ |
| Monitor (exit) | | ✗ |