Universidade do Minho

Mestrado Integrado em Engenharia Informática

Gestão de Vendas

Sistemas Operativos

2º Ano, 2º Semestre, 2018/2019

Grupo 5

84668 - Francisco Peixoto
86268 - Maria Pires
85242 - Maria Regueiras

11 de maio de 2019

Conteúdo

Lista de Figuras

Capítulo 1

Introdução

Este relatório contém a apresentação do projeto desenvolvido no âmbito da disciplina de *Sistemas Operativos*. Serão expostos os problemas que o grupo confrontou e as soluções implementadas para a construção de um sistema de gestão de inventário e vendas.

Para conseguir implementar este projeto recorremos a conceitos apresentados nas aulas da unidade curricular, tais como as *system calls*: *pipes*, *fork*, *exec*, *read*, etc, tendo sempre em consideração o melhor desempenho possível dos programas.

Capítulo 2

Resolução do Problema

Para a resolução do problema proposto começamos por esquematizar a arquitetura do serviço de gestão de vendas e as interações entre cada módulo do sistema, conforme apresentado na figura seguinte.

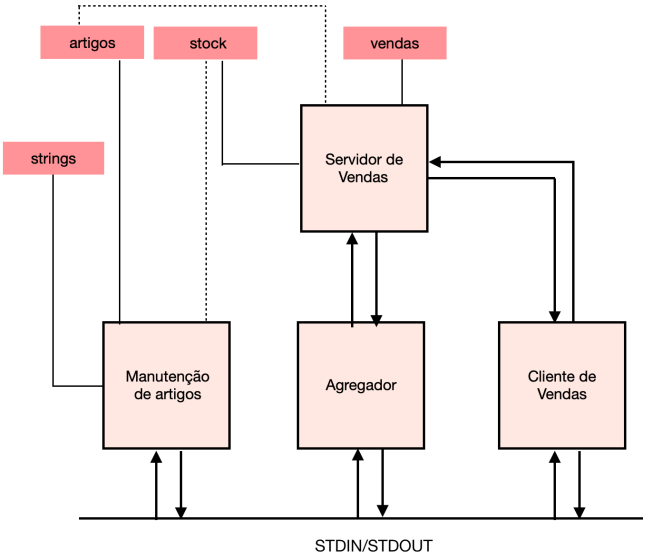


Figura 2.1: Funcionamento do sistema

As setas da figura representam as ligações, tais como pipes, que foram implementadas para que os programas pudessem comunicar entre si. A escolha da arquitetura foi também afetada pelos seguintes fatores:

- Concorrência: Um dos maiores desafios deste projeto foi implementar de forma correta e eficiente os módulos que executam concorrentemente.
- Sincronização: Se dois programas tentarem ler um ficheiro em comum não há problema, contudo se ambos tentarem escrever ao mesmo tempo para esse ficheiro podemos ter inúmeros problemas pelo que quando um processo está a escrever tivemos em atenção o acesso exclusivo aos ficheiros.

Capítulo 3

Componentes do projeto

3.1 Manutenção de Artigos

A manutenção de artigos é a parte do programa que permite a inserção de artigos no inventário e a alteração das suas características. Inicialmente foram implementadas as funções que permitem a inserção de um artigo no ficheiro *artigos*, onde é escrita a referência e o preço de cada um em binário. Optamos por guardar também o tamanho do nome do artigo, de modo a facilitar o cálculo da sua posição no ficheiro *strings*. Este acede ainda ao ficheiro *stocks* para acrescentar o stock do artigo a 0.

Formato do ficheiro *artigos*:
<referencia> <tamanho> <preço>

O código de cada artigo não se encontra no ficheiro por uma questão de poupança de memória visto ser desnecessário. O código é um inteiro que corresponde ao index do artigo no ficheiro, gerado ao mesmo tempo que é inserido. Isto é, se quiser aceder ao artigo 50, apenas é necessário multiplicar 49 (pois os códigos começam no 1) pelo tamanho de um artigo (referência, tamanho e preço).

A referência, por sua vez é a posição (em bytes) onde começa o nome do artigo, que se encontra no ficheiro *strings*.

Formato do ficheiro *strings*:
<nome>

É também possível alterar o preço e o nome de um artigo, a alteração de preço apenas afeta um ficheiro, contudo a alteração de um nome é feita adicionando ao ficheiro *strings* o novo nome e gerando uma nova referencia que substitui no ficheiro *artigos* a referencia antiga. O nome antigo permanece no ficheiro *strings* até que o espaço desperdiçado ultrapasse 20% da capacidade total do ficheiro (compactação do ficheiro strings).

3.2 Servidor de Vendas

O servidor de vendas é responsável pelo controlo dos stocks e registo das vendas. Este recebe pedidos dos outros módulos, emite respostas, e comunica com quase todos os restantes módulos, sendo por isso o módulo central do projeto.

Formato do ficheiro *stocks*:
<quantidade em stock>

O servidor tem acesso aos ficheiros *stock* e *vendas* e, para além destes, têm ainda acesso ao ficheiro *artigos* (apenas para consulta de preços).

Formato do ficheiro *vendas*:
<código do artigo> <quantidade vendida> <montante total>

Neste ficheiro é incluído o código, pois quando se insere uma venda esta é adicionada ao fim do ficheiro, sendo esta a única maneira de identificar o artigo vendido. Para além disso, inclui-se a quantidade vendida e o montante total correspondente.

Pedidos efetuados pelos outros módulos:

- Cliente de Vendas: pede ao servidor para acrescentar uma venda no ficheiro, acrescentar no stock e consultar o stock e o preço de um determinado artigo.
- Agregador: pede dados da última agregação.

O servidor de vendas é ainda capaz de correr o agregador, quando solicitado pelo ma.

3.3 Cliente de Vendas

Este módulo apenas comunica com o servidor de vendas, através de *named pipes*, podendo ser efetuados pedidos de vários clientes que são atendidos pelo servidor, conforme demonstrado na figura seguinte:

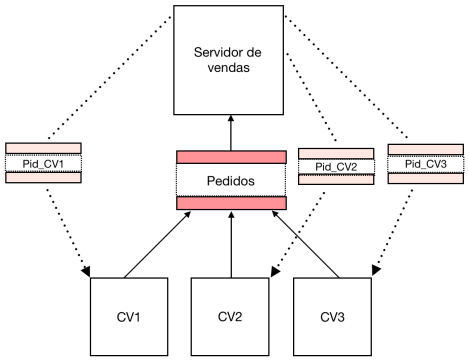


Figura 3.1: Comunicação entre SV e CV

Uma particularidade deste módulo é a capacidade de múltiplos clientes executarem em paralelo, assim, foi criado um pipe geral *pedidos*. Este pipe recebe os pedidos de todos os clientes, contudo, para identificar o cliente a que lhe pertence primeiro é enviado o seu pid. Após o processo do pedido por parte do servidor, este envia a resposta para um pipe individual criado pelo cliente cujo nome é o seu próprio pid.

3.4 Agregador

O agregador funciona isoladamente não tendo conhecimento de quaisquer dados para além dos fornecidos pelo stdin. Inicialmente foi implementado sequencialmente, comparando por linha as vendas, o que para além de ser extremamente ineficiente era também muito lento. Foi então feito um agregador concorrente.

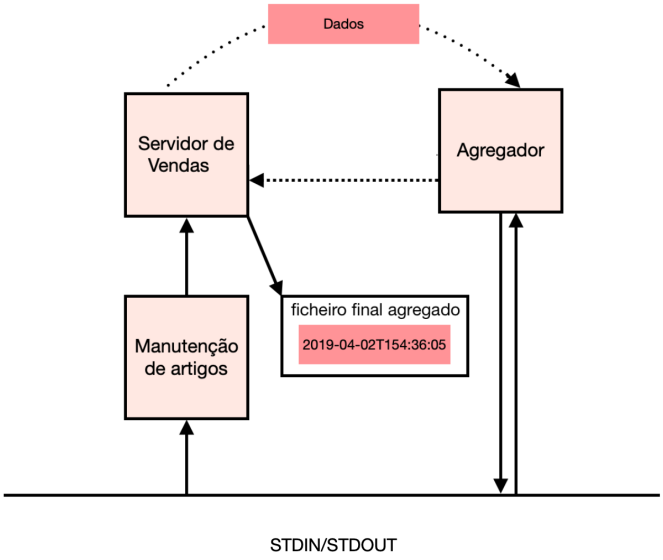


Figura 3.2: Funcionamento do agregador

3.5 Componentes adicionais

3.5.1 Agregação concorrente

De modo a obter uma agregação concorrente procedeu-se à divisão do ficheiro de vendas em n partes, sendo estas atribuídas a n filhos que agregam pequenas partes do ficheiro concorrentemente.

3.5.2 Compactação do ficheiro Strings

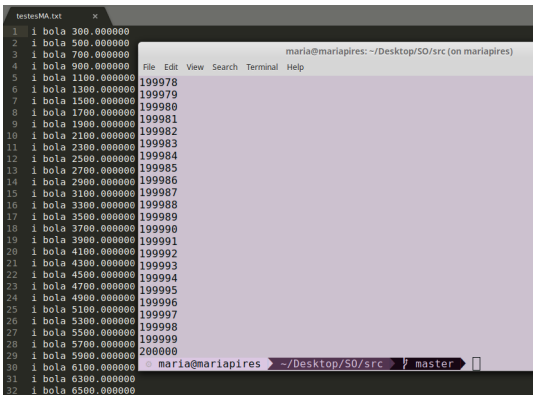
Quando é requisitada a mudança de nome de um artigo é adicionado um novo nome ao ficheiro *strings*, o nome antigo permanece no ficheiro havendo uma acumulação de lixo. Para lidar com esta situação cada vez que há uma alteração, os tamanhos dos nomes antigos são somados e guardados no inicio do ficheiro,

de modo a poder recuperar o total de lixo acumulado. O lixo é comparado com o tamanho total do ficheiro, assim que este ocupa 20% é invocada a função *clearTrash* que obtém o offset de cada nome do ficheiro *strings* através das referencias do ficheiro *artigos*, caso haja correspondência o nome é adicionado a um novo ficheiro, sem nomes obsoletos, e as referencias e tamanho são atualizados no ficheiro *artigos*.

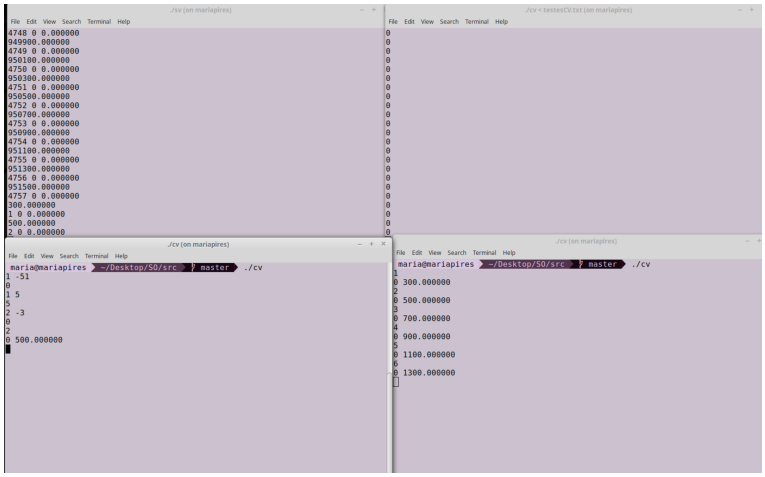
Capítulo 4

Testes

Para testar o servidor foram criados ficheiros de teste, tais como o testesMA.txt,



Testamos ainda os pedidos de múltiplos clientes em simultâneo ao servidor de vendas.



Capítulo 5

Conclusão

O trabalho foi realizado com sucesso apesar de o grupo ter encontrado alguns contratempos. Houve bastante dificuldade na gestão da sincronização, deparámo-nos com um *deadlock* na implementação do servidor de vendas concorrente, contudo fomos capazes de o corrigir e prosseguir.

A cache foi o único aspeto do enunciado que não foi implementado, pois embora fosse bastante fácil e já estivesse planeada a criação de um array FIFO para funcionar como uma cache dos artigos mais recentes os problemas de sincronização impediram-nos de continuar.

O grupo avalia o trabalho realizado duma forma bastante positiva no que respeita à compreensão dos conceitos abordados e à sua aplicação.