Projeto (Fase I)

1 Introdução ao Projeto

A parte teórico-prática da disciplina de Sistemas Operativos pretende familiarizar os alunos com alguns dos problemas envolvidos na utilização dos recursos de um Sistema Operativo. O projeto de avaliação será realizado utilizando principalmente a linguagem de programação C e as APIs de Linux e POSIX (*Portable Operating System Interface*), mas inclui também alguma programação de em `shell` (`bash` e `makefile`).

O propósito geral do projeto é o desenvolvimento, de forma modular e faseada de uma aplicação em C que criará múltiplos processos cooperativos que irão simular o fluxo central de um serviço de vendas online (SO_Store). Este fluxo envolve a realização de encomendas e emissão de recibos deixando de fora, por simplificação, o fluxo de pagamento. De forma a se poder aferir a qualidade do serviço são registadas informações de progresso (log) que podem posteriormente ser analisadas.

O projeto SO_Store será realizado em 5 fases. A primeira fase tem como objetivo fundamental a familiarização com o código existente e a criação do ficheiro `makefile` para compilação e manutenção do projeto. Neste primeiro enunciado é feita uma apresentação geral da SO_Store juntamente com informação específica de suporte à realização da primeira fase. Em cada fase seguinte será disponibilizado um novo enunciado que complementará a informação contida nos anteriores.

2 Funcionamento Geral

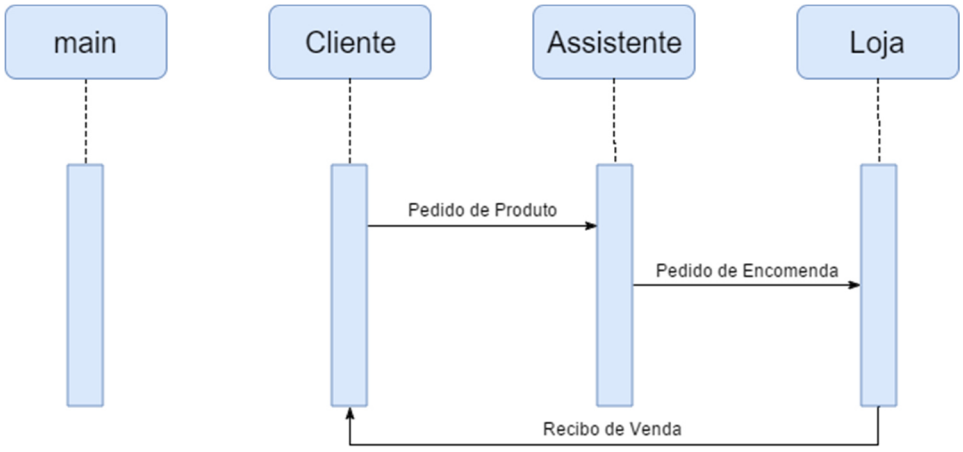

Figura 1 - Diagrama de sequência para a venda de produtos na SO_Store

A SO_Store agrega informação sobre um catálogo de produtos disponível através de várias lojas, facilitando a compra dos mesmos através da utilização de uma assistente digital. O fluxo central do sistema a simular (Figura 1) compreende as seguintes etapas:

1. Um cliente da SO_Store, consulta a lista de produtos e preenche o formulário de pedido de produto, indicando qual o produto que pretende, e coloca-o no cesto de pedidos de produtos.
2. Uma assistente digital obtém o próximo pedido de produto do cesto respetivo, verifica a disponibilidade em stock desse produto e, caso esteja disponível, preenche um pedido de encomenda que coloca no cesto de pedidos de encomenda.

3. Uma loja obtém o próximo pedido de compra do cesto respetivo, efetua o empacotamento da mesma e preenche o recibo para o cliente, colocando-o no cesto de recibos.
4. Um cliente cujo produto tenha sido encomendado com sucesso faz download do seu recibo e termina a sua interação com o sistema. Posteriormente irá receber o seu produto entregue por um drone o mais rápido possível.

3 Interação com o Sistema Operativo

Na SO_Store as entidades (clientes, assistentes, lojas) são representadas por processos que cooperam através de cestos (buffers) para troca de pedidos/documentos.

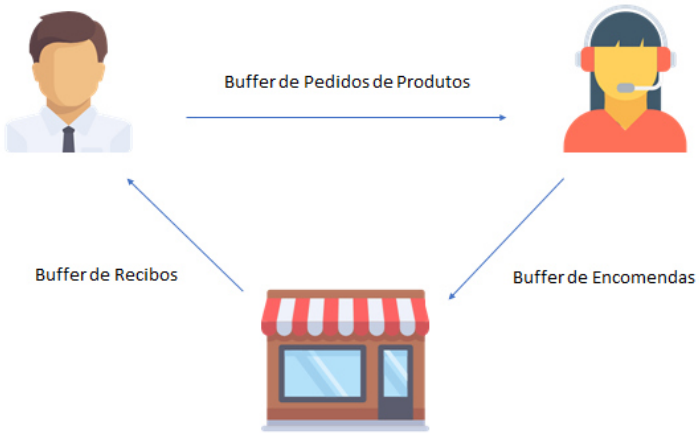

Figura 2 - Comunicação entre processos através de memória partilhada

Os processos comunicam através da escrita e leitura em buffers que não são mais do que zonas de memória com uma determinada capacidade para guardar estruturas de dados de um dado tipo (vetor de estruturas). Cada estrutura de dados representa um pedido, uma encomenda ou um relatório, consoante o buffer. Uma vez que um processo não pode aceder ao espaço de endereçamento de outros processos, estes buffers têm de ser concretizados através de **zonas de memória partilhada**.

De modo a que o acesso às zonas de memória partilhada seja coerente, os processos terão de se **sincronizar**. Adicionalmente, os vários processos também necessitam de se coordenar na realização das suas atividades.

A configuração do sistema e a escrita de resultados é feita através de **ficheiros**. Desta forma a configuração pode ser feita sem necessidade de recompilação da aplicação e os resultados permanecem guardados após a execução do SO_Store.

A execução é acompanhada através da atualização de **relógios** que permitem registar o estado. Os estados são armazenados em ficheiro para posterior análise. Adicionalmente através de **alarmes** é possível mostrar regularmente no ecrã informação sobre a evolução do sistema.

Finalmente, a organização e manutenção dos ficheiros do projeto é feita através de **shell scripts**. A primeira fase do projeto vai debruçar-se sobre este último tema. Os restantes temas mencionados anteriormente serão explicados em mais detalhe nos próximos enunciados.

4 Configuração

A execução da SO_Store é parametrizada através de um ficheiro de configuração que descreve um determinado cenário. Em cada cenário são descritos precisamente quais os produtos disponibilizados, o nº de clientes e os produtos que querem comprar, o nº de assistentes digitais, o número de lojas e os produtos oferecidos por estas, e finalmente a capacidade dos diversos cestos de pedidos/encomendas/recibos

(buffers). Esta abordagem permite testar facilmente o sistema em diferentes cenários. Um cenário possível é o seguinte:

```
; tamanho máximo da linha = 500 chars

[PRODUTOS]
; máximo de produtos = 10
; considera-se que os produtos são identificados sequencialmente por (0 - 9)
; o stock de cada produto é indicado pela quantidade (0 - ...)
; neste caso são definidos os stocks de 6 produtos numerados de 0 a 5.
STOCK = 1 2 3 4 5 6

[CLIENTES]
; o numero de clientes é dado pelo número de elementos na lista
; os clientes não têm identificadores
; o produto requisitado por cada um é indicado pelo número (0 - 9)
PRODUTO = 0 1 2 3 4 5

[ASSISTENTES]
; o número de assistentes é dado pelo número de elementos da lista
; cada assistente é identificada por (A1, A2, ...)
LISTA = A1

[LOJAS]
; o número de lojas é dado pelo número de elementos da lista
; os produtos que cada loja disponibiliza são indicados pelos números (0 - 9)
; os produtos de cada loja são separados por virgulas
; neste caso existem 2 lojas que vendem qualquer um dos produtos
ESPECIALIDADES = 0 1 2 3 4 5, 0 1 2 3 4 5

[BUFFERS]
; o número de buffers não pode ser alterado
; cada buffer é caracterizado por uma capacidade que indica o número máximo de
; pedidos/encomendas/recibos que podem ser armazenados num dado instante
; os valores da capacidade podem ser alterados
CAPACIDADE = 10 10 10
```

5 Estrutura do projeto

Os ficheiros do projeto podem ser descarregados da página da disciplina. Depois de descomprimido o arquivo, serão encontrados os seguintes diretórios:

```
\so_store
      \bin
      \include
      \logPlayer
      \obj
      \src
      \testes
            \in
            \log
            \out
```

O diretório bin contém o executável `sostore`. O diretório `include` contém os ficheiros `.h` com a definição das estruturas de dados e declarações de funções. Estes ficheiros não podem ser alterados. O diretório `logPlayer` contém o ficheiro fonte do depurador, `logPlayer`, que será apresentado na secção seguinte. O diretório `obj` contém os ficheiros objeto. O diretório `src` contém os ficheiros fonte que deverão ser alterados para realização do projeto. Finalmente o diretório `testes` inclui os vários ficheiros de configuração (em `in`) da SO_Store e é também onde devem ser guardados os ficheiros gerados (`out` e `log`). O executável `sostore` será gerado a partir da execução do comando `make`. O `makefile` necessário para a execução do comando `make` deve ser colocado no diretório `so_store`.

6 Output

Quando a SO_Store é executada com um determinado ficheiro de configuração, são gerados dois ficheiros: um ficheiro de log e um ficheiro de output. O `logPlayer` permite ver o conteúdo (ou parte) de um ficheiro de log que pode ser gerado pelo sistema. Este ficheiro de log pode der muito útil para fazer depuração do programa por mostrar o conteúdo dos 3 buffers de comunicação (cliente, assistente e loja) sempre que são alterados. Para gerar um ficheiro de log com a SO_Store deve ser utilizada a opção "`-l`" seguida do nome do ficheiro de log a criar.

O ficheiro de log é constituído por registos separados por uma sequência de 4 inteiros com todos os bits a um. Cada registo contém o tempo decorrido (`double`), a etapa em que foi gerado (`int`), o id de quem o gerou (`int`) (cliente, assistente ou loja) e o conteúdo das posições ocupadas dos 3 buffers, respetivamente: `BProduto`, `BEncomenda` e `BRecibo` (múltiplos de 4 `int`).

A etapa em que foi gerado é um numero natural cujo valor corresponde, respetivamente às seguintes etapas no acesso aos buffers:

1. Cliente escreve pedido de produto;
2. Assistente lê pedido de produto;
3. Assistente escreve pedido de encomenda;
4. Loja lê pedido de encomenda;
5. Loja escreve recibo;
6. Cliente lê recibo.

O conteúdo de cada buffer está separado por uma sequência de 2 inteiros com todos os bits a 1 e mais 2 inteiros com todos os bits a 0. O conteúdo de cada buffer é constituído por 0, ou mais, entradas de 4 `int` cada. Cada entrada contem 4 identificadores que são: id do produto (`int`), id do cliente (`int`), id da assistente (`int`) e id da loja (`int`). O utilitário pode ser chamado passando apenas o nome do ficheiro de log a mostrar. Podem também ser especificados, na linha de comando, os seguintes filtros: etapa a visualizar e/ou id do cliente/assistente/loja a visualizar e/ou o número do primeiro registo a mostrar e/ou o número do último registo a mostrar. Considera-se que os registo a visualizar são numerados sequencialmente a partir de 1, em dependência dos filtros escolhidos. Para conhecer as opções disponíveis basta executar o `logPlayer` sem parâmetros. Os ficheiros de log deveram ser colocados no diretório `so_store/testes/log`.

Os ficheiros de output da SO_Store deveram ser colocados no diretório `so_store/testes/out`. Estes ficheiros vão ter um aspeto que vai depender do cenário/configuração inicial, mas serão semelhantes ao seguinte:

```
CLIENTE 000 encomendou 0 e obteve 0
CLIENTE 001 encomendou 1 e obteve 1
CLIENTE 002 encomendou 2 e obteve 2
CLIENTE 003 encomendou 3 e obteve 3
CLIENTE 004 encomendou 4 e obteve 4
CLIENTE 005 encomendou 5 e obteve 5
Clientes: 06
```

```
Assistentes: 01
Lojas: 01
Clientes atendidos pelas assistentes:      06
Clientes atendidos pelas lojas:            06
Total vendas:                         01 01 01 01 02 00 = 06
Stock inicial:                        01 02 03 04 05 06 = 21
Stock final:                          00 01 02 03 04 05 = 15
```

Estes ficheiros sumarizam a execução do cenário/configuração dado como input e mostram estatísticas importantes e acontecimentos que decorreram ao longo da execução do programa.

7 Teste

O teste do código que será futuramente elaborado por cada grupo será feito por comparação de um dado ficheiro de output gerado por esse código com o ficheiro de output gerado pelo `sostore` padrão (disponibilizado no pacote inicial).

8 Objetivos

Esta primeira fase do projeto tem como objetivo a familiarização dos alunos com o projeto e criação do ficheiro `makefile` que permitirá a compilação automática do mesmo. Para tal devem explorar o código fornecido e testar diferentes ficheiros de configuração como input.

O ficheiro `makefile` deve conter pelo menos:

1. Regras de compilação para os ficheiros objeto;
2. Regra de ligação para o executável `sostore`, incluir bibliotecas -lrt e -lpthread
3. Regras de limpeza de projeto (`clean`) que devem limpar a diretoria de ficheiros objeto e o executável `sostore` (Cuidado para não remover o ficheiro so.o fornecido pelos docentes);
4. Regras de teste.

Para o ponto 4 deve ser escrito um *shell script* (soXXX.sh em que XXX é o número do grupo) que ordena e compara os ficheiros de output gerados pelo código fornecido com os ficheiros de output gerados pelo código elaborado e escreve no terminal "IGUAIS" caso os ficheiros sejam iguais ou "DIFERENTES" caso contrário. O *shell script* deve receber os nomes dos ficheiros a testar como parâmetros. Como nesta fase apenas existe o `sostore` padrão, então é preciso "criar" outros ficheiros de output para teste do *shell script*. Estes podem ser criados partindo de um ficheiro de output gerado pelo `sostore` padrão, efetuando as seguintes alterações:

1. Troca da ordem das linhas do ficheiro (o resultado do teste deverá ser "IGUAIS")
2. Alteração de valores no ficheiro (o resultado do teste deverá ser "DIFERENTES")

9 Entrega

Os ficheiros `makefile` e `soXXX.sh` devem ser entregues até às 23h do dia 19/03/2017, utilizando o link de entrega disponível na página de SO no moodle.

Não serão aceites trabalhos por e-mail nem por qualquer outro meio não definido nesta secção. Se não se verificar algum destes requisitos o trabalho é considerado não entregue.