



SOBBay



Trabalho Prático de Sistemas Operativos – Meta 2

Renato Alexandre Oliveira Craveiro – 2018011392

Sistemas Operativos | Ano Letivo: 2022/23

"In theory there is no difference between theory and practice - in practice there is" (Yogi Berra)

Índice

Introdução.....	3
Estrutura do projeto.....	4
Biblioteca <i>SOBay.h</i>	4
Ficheiros internos do projeto / Ficheiros temporários gerados pelo mesmo.....	4
Programa <i>./Frontend</i>	5
Programa <i>./Backend</i>	5
Programa <i>./Promotor</i>	5
Bibliotecas / programas externos utilizados.....	6
Makefile do projeto.....	6
Implementação.....	7
Conclusão e dificuldades	8

Introdução

Este projeto consiste na criação de um sistema de venda por leilão de itens, tal como o já conhecido eBay.

Vários utilizadores da plataforma vão querer vender e comprar itens, itens esses que estarão disponíveis na plataforma por um dado período (representado em segundos).

Cada item tem um nome, uma categoria, um preço base (que poderá ser alterado caso outro utilizador faça uma licitação maior), um preço “compre já” (como o nome descreve, preço ao qual o produto fica vendido automaticamente) e um “tempo de vida” (findo este tempo, o item deixa de estar disponível na plataforma).

Poderão ser lançadas promoções de categorias que farão baixar os preços de itens pertencentes a uma categoria durante um certo período.

A gerência dos utilizadores, itens, promoções e funcionalidades do sistema é feito pelo programa *backend*, sendo este operado pelo administrador.

Os utilizadores interagem com o sistema a partir do programa *frontend*. Através deste inserem comandos que serão encaminhados para o *backend* e receberão respostas do mesmo.

Todos os utilizadores são avisados e encerram caso o *backend* encerre.



```
renato@renato-OMEN-by-HP-Laptop-15-dcxxxx: ~/Documents/50/TrabPrat22_23/so-2223-tp-m2-RenatoCraveiro
renato@renato-OMEN-by-HP-Laptop-15-dcxxxx: ~/Documents/50/TrabPrat22_23/so-2223-tp-m2-RenatoCraveiro$ ./Backend
[SEN VAR AMBIENTE HEARTBEAT! APLICAR DEFAULT 60]
[HEARTBEAT de cada utilizador: 60]
[6 utilizadores em sistema]

[Tempo em execução: 1257 seg.]

[4 itens em sistema]
LAST ITEM ID 7

[3 promotores em sistema]:

Command:
ya Entrou!
Um utilizador tentou entrar com User/pass inválida!
user Salu!
nome Entrou!
RECEBIDO HEARTBEAT DE ya

[BACKEND] TEH 54787 DE SALDO DISPONIVEL
```

Backend e 2 Frontends em execução

Estrutura do projeto

O projeto SOBay divide-se em várias partes, sendo estas organizadas por diretorias:

- A diretoria *backend* contém os ficheiros de código relativos ao programa *./Backend*
- A diretoria *frontend* contém os ficheiros de código relativos ao programa *./frontend*
- A diretoria *promotor* contém o ficheiro de código do promotor desenvolvido durante esta meta, como os restantes promotores disponibilizados pelos docentes da Unidade Curricular para o desenvolvimento do projeto
- A diretoria *SOBay_common* contém os ficheiros de código e biblioteca utilizados pelo *./Backend* e pelo *./Frontend*, contendo estruturas e funções comuns em ambos
- A diretoria *files* contém os ficheiros de utilizadores, itens e promoções para o armazenamento, gestão e manipulação destas mesmas informações

```
/SOBay$ tree
.
├── backend
│   └── backend.c
├── files
│   ├── item
│   ├── promo
│   ├── tempo
│   └── users
├── frontend
│   └── frontend.c
├── makefile
├── promotor
│   ├── black_friday
│   ├── promotor
│   ├── Promotor
│   ├── promotor.c
│   └── promotor_oficial
└── SOBay_common
    ├── soBay.c
    ├── soBay.h
    ├── users_lib.h
    └── users_lib.o
```

Estrutura de árvore do projeto

Biblioteca SOBay.h

Localiza-se na diretoria *SOBay_common* e contém as estruturas e funções utilizadas pelos *./Promotor* e *./Backend*.

Esta contém:

- Todos os *#includes* de bibliotecas necessárias ao projeto (requeridas pelos programas envolvidos).
- Definições de variáveis globais de como *MAXUSERS* (que indica o nº máximo de utilizadores que o projeto suporta), *MAXPROMO* (que indica o nº máximo de promotores em execução), *MAXITEMS* (que indica o nº máximo de itens em leilão), as variáveis *FUSERS*, *FPROMOTERS* e *FITEMS* que indicam, respetivamente, a localização dos ficheiros de utilizadores, promotores e itens, entre outras
- Estruturas que definem os itens, leilões e promoções. Ainda existe uma estrutura *command* que contém dados para interpretação de comandos inseridos por utilizadores/administradores do projeto nas várias aplicações.
- Ainda contém a função *splitstring* – que recebe uma string e devolve uma estrutura *command* com o comando separado dos parâmetros, indicando também o número de parâmetros detetados.

Ficheiros internos do projeto / Ficheiros temporários gerados pelo mesmo

Existem 3 ficheiros localizados na diretoria *files*:

- users: guardam informação relativa aos utilizadores (*username password saldo\n*)
- item: guardam informações relativamente os itens
- promo: guardam os nomes dos programas promotores
- tempo: que guarda o número de segundos registados até ao momento.

O programa *./Backend* ainda gera um ficheiro temporário no arranque (*be.init* guardado na diretoria em que o *./Backend* se encontra executado), que serve para controlar se outra instância se encontra em execução (ou não). Ao encerrar, este ficheiro é eliminado.

Programa *./Frontend*

Este programa tem como finalidade receber dados dos utilizadores do projeto. Nesta fase, este programa faz:

- receção de credenciais ao ser “lançado” (*./Frontend [username] [password]*) (sendo esta validação feita pelo *backend*, que responderá em conformidade: se for correto continua a execução, se não termina)
- receção de comandos e resposta/validação de parâmetros

Programa *./Backend*

Este programa tem a finalidade de administrar todo o projeto e gerenciar utilizadores, itens e promoções. O programa *./Backend*, nesta fase:

- Tal como referido anteriormente, o *./Backend* cria um ficheiro para que, caso outro seja lançado, saiba que já existe um em execução
- Durante a execução, o mesmo pergunta que tipo de operação deseja efetuar: comandos, utilizadores, promoções ou itens
- Interpreta comandos escritos pelo Admin – executando-os apenas quando os mesmos são válidos
- Em termos de opções de promotor, o *./Backend* permite listar promotores listados no ficheiro de promotores, executar o primeiro promotor da lista e cancelar todos os promotores em execução
- A opção de itens apenas permite listar os itens listados no ficheiro
- As operações de utilizadores retiram sempre um de saldo a todos os utilizadores, apenas ao pretender utilizar uma destas operações (ou seja, ao escolher a opção utilizadores no menu anterior). Permitem listar todos os utilizadores presentes no sistema, testar um login de user, ver o saldo de um determinado utilizador, e atualizar o saldo de um determinado utilizador.

Estes 3 últimos pontos podem ser acedidos através do comando *“others”*.

Programa *./Promotor*

O *./Promotor* é um promotor criado para este projeto que imprime entre tempos aleatórios (em segundos) promoções das categorias de construção, informática, *gaming*, literatura e escritório, com valores e tempos aleatórios. Encontra-se na diretoria promotor.

Bibliotecas / programas externos utilizados

Em termos de bibliotecas e programas externos, estas foram fornecidas pelos docentes da Unidade Curricular. São utilizados os programas “*promotor_oficial*” e “*black_friday*” (localizados na diretoria *promotor* e a biblioteca “*users_lib.h*” (localizada na diretoria *SOBay_common* juntamente com o binário “*users_lib.o*” correspondente).

Makefile do projeto

Para facilitar a compilação de todos os programas envolvidos, foi criado um makefile, que permite, com o comando *make all* compilar os programas *./Backend* e *./Frontend*. Estes utilizam ficheiros binários (.o) para ajudar na compilação e juntar com as bibliotecas presentes no projeto. Para remover os ficheiros compilados, utiliza-se o comando *make clean* (o comando *make cleanall*, também elimina o *./Promotor*).

Implementação

As funcionalidades do *Backend* de leitura de mensagens de utilizadores, de atualização de tempo (e atualizações de estruturas que alteram com o tempo) e de leitura de comandos do Administrador (através do teclado) são executadas em paralelo (simultaneamente) através de *Threads POSIX*.

A interação entre *Backend* e *Frontend* é feita através de *named pipes*. Existe um *named pipe* do *Backend* que recebe todas as interações de todos os utilizadores, sendo que o backend interpreta as mensagens e responde de volta através do *named pipe* associado ao *Frontend* que enviou a mensagem. Os *frontends* podem enviar *heartbeats*, comandos, indicações de login e indicações de *logoff*, e o *Backend* responde com as mensagens de comando sucedido (com o respetivo output do comando) ou comando não sucedido, reagem ao *heartbeat* atualizando o tempo de *heartbeat* restante, indicam um login sucedido ou não e enviam informações de *kick* a um *user* ou de encerramento do *Backend* a todos os *users* conectados. Ainda enviam informações de novos itens, itens que foram vendidos ou terminaram o tempo sem venda e início e fim de promoções aos *users* online.

O *Frontend* executa simultaneamente a leitura do teclado e a recepção de mensagens do *Backend* através de *threads* POSIX, reagindo ao tipo de mensagem em conformidade (se for um *kick* encerram, se for o encerramento do *Backend* encerram ou se for a resposta a um comando enviado imprimem a resposta do *Backend*).

Os promotores são lançados através do *Backend*. Nesta implementação apenas um promotor pode ser lançado de cada vez, tendo que se esperar pela promoção ser lançada. Quando rececionada, a promoção é registada e aplicada aos itens correspondentes.

Ao encerrar, o *Backend* grava todos os dados de *users*, *itens* e tempo nos ficheiros correspondentes.



Backend em execução com a inicialização das várias estruturas e receção do comando "list"

Conclusão e dificuldades

O desenvolvimento deste projeto foi desafiante, tanto na implementação, como na descoberta de funcionalidades de sistema e comunicação entre processos.

Consegui compreender como funciona a comunicação entre processos, como funcionam *threads* e a importância das mesmas para a execução de várias partes do código de um processo em “simultâneo”, como o Sistema Operativo consegue gerir um processo e a execução/término do mesmo e as ferramentas que o mesmo oferece para a obtenção de dados.

Durante o processo de desenvolvimento encontrei várias dificuldades, mas, infelizmente, não consegui ultrapassar algumas, como o lançamento de promoções, no entanto o resto do projeto encontra-se “minimamente funcional” (na minha opinião).

Foi um projeto bastante interessante pois, normalmente, este tipo de projetos é projetado para a web, e desenvolvê-lo numa “outra perspetiva” (de terminal/comandos de linha) foi desafiante de imaginar, mas “engraçado” de ver o resultado.

Para concluir, foi um projeto que gostei imenso de desenvolver, mesmo que não esteja “a 100%”.



©Renato Craveiro | 2018011392 | LEI-PL | ISEC 2022/23 | Sistemas Operativos
<https://github.com/renato-craveiro/>
<https://gamejolt.com/@renatocraveiro>

“Banner” do projeto “SOBay” e créditos