

MC833

Projeto 1 - Servidor concorrente sobre TCP

Introducao

O objetivo deste projeto é implementar um servidor concorrente sobre TCP, num contexto de catálogo de filmes: o servidor deve permitir que clientes conectados a ele executam uma série de operações.

O código desenvolvido foca nos conceitos relevantes ao curso: comunicação em rede, no modelo cliente-servidor. Toda a parte de processamento das operações e armazenamento dos dados dos filmes foi implementada de forma a ser simples, sem depender de ferramentas de serialização e nem mesmo de *parsing* complexo de dados. Esses detalhes são esclarecidos nas seções a seguir.

Descricao geral e casos de uso

Ao se conectar ao servidor, um cliente pode executar as seguintes operações:

- cadastrar um novo filme recebendo como resposta positiva o seu respectivo identificador;
- remover um filme a partir do seu identificador;
- listar o título e salas de exibição de todos os filmes;
- listar todos os títulos de filmes de um determinado gênero;
- dado o identificador de um filme, retornar o título do filme;
- dado o identificador de um filme, retornar todas as informações deste filme;
- listar todas as informações de todos os filmes.

O servidor processa e armazena os títulos, sinopses, gêneros, salas em exibição, e atribui um identificador único para cada filme armazenado.

Armazenamento e estrutura de dados do servidor

Todos os dados dos filmes são armazenados em arquivos. Como mencionado na introdução, para manipular e armazenar os dados dos filmes o servidor não faz uso de nenhuma ferramenta de serialização e nem mesmo precisa processar os dados ao lê-los dos arquivos.

Isso é possível graças a um mapeamento de um para um dos arquivos com cada campo de dado de cada filme, realizado da seguinte forma: são criadas pastas para cada campo de dados dos filmes e nessas pastas é criado um arquivo para cada filme (os arquivos são nomeados com o sufixo “_<id>”, onde *id* é o identificador do filme)

Assim, na pasta *titles/* são encontrados os arquivos com nome “*title_<id>*” e assim por diante para as pastas *synopses/*, *genres/* e *rooms/*.

Com isso, para obter o título do filme com *id* = 2, por exemplo, basta ler o arquivo *titles/title_2*.

Detalhes de implementacao

O cliente e servidor “concordam” em enviar sempre a mesma quantidade x de bytes. Isso é forçado pela diretiva “`#define MAX_MSG_SIZE 10000`” no arquivo `utils.h`. Essa concordancia serve para facilitar na hora de receber mensagens, de ambos os lados: quem esta recebendo sempre sabe o quanto esperar de quem esta enviando e, por isso, consegue parar de receber sempre quando todos os dados que eram para ser recebidos foram recebidos.

Isso fica bem claro no codigo, acompanhado de comentarios que explicam como isso ocorre.

Para execucao do codigo

Para executar o servidor, compile:

```
gcc server.c -o server
```

E execute:

```
./server
```

Para executar o cliente, compile:

```
gcc client.c -o client
```

E execute:

```
./client <hostname>
```

onde *hostname* é o output do comando Linux *hostname*.

A partir daí basta seguir o que aparece no terminal.

Conclusao

O servidor implementado atende todos os requisitos de operações do projeto.