Instituto Superior de Engenharia

Politécnico de Coimbra

Departamento de Engenharia Informática e de Sistemas

Sistemas Operativos

2024/2025

PROGRAMAÇÃO EM C PARA UNIX

SISTEMA DE MENSAGENS

Chelsea Carina Pais da Cunha Duarte – 2021100010

Rodrigo Miguel Pessoa da Bernarda – 2021136740

Coimbra

2024

INDÍCE

ÍNDICE DE TABELAS E FIGURAS

1 INTRODUÇÃO

Este trabalho tem como objetivo a criação de um sistema de mensagens, composto por dois programas principais: o "manager" e o "feed". O sistema permite que vários utilizadores se conectem ao servidor (manager) para enviar e receber mensagens de tópicos específicos subscritos por cada utilizador.

O sistema foi desenvolvido no IDE Visual Studio Code, utilizando a linguagem C, para a plataforma UNIX (Linux), dentro de um ambiente virtual. Foram aplicados conceitos, mecanismos e recursos do sistema lecionados na unidade curricular de Sistemas Operativos, do Instituto Superior de Engenharia de Coimbra, tais como gestão de processos, comunicação entre processos por meio de FIFOs, sinais e sincronização de threads.

Neste relatório, são explicadas a arquitetura do sistema, as decisões tomadas durante a implementação, e as práticas seguidas o desenvolvimento do sistema. Por fim, são apresentadas as funcionalidades desenvolvidas e as conclusões.

2 FUNCIONAMENTO E DESCRIÇÃO GERAL DO PROJETO

O projeto consiste na implementação de um sistema de envio e receção de mensagens curtas, que permita aos utilizadores, através do programa *feed*, enviar e receber mensagens relacionadas a tópicos específicos, determinados por cada utilizador. O sistema é composto por dois componentes principais: o **manager**, que funciona como servidor central, e o **feed**, que trabalha como interface para os utilizadores interagirem com o sistema.

2.1 Funcionamento do Sistema

O sistema permite que os utilizadores se conectem e interajam com o sistema através de comandos de texto. A seguir, será descrito o funcionamento geral do sistema e os pormenores sobre as decisões tomadas para o funcionamento das suas funcionalidades.

2.1.1 Conexão e Validação do Utilizador

Para que um utilizador se conecte ao sistema, é necessário que ele indique o seu **nome de utilizador** através da execução do programa *feed* pela linha de comandos. No entanto, a conexão só será estabelecida se não ocorrer nenhuma das seguintes condições:

1. **Nome de utilizador não indicado**: O nome do utilizador deve ser sempre indicado para validação, como um procedimento de login;
2. **Manager não disponível**: Caso o servidor (manager) não esteja em execução, o utilizador não conseguirá aceder ao sistema;
3. **Validação do utilizador falhada**: Se o nome de utilizador já estiver em uso ou se o limite máximo de utilizadores já tiver sido atingido, a validação feita pelo manager falha, e o acesso ao sistema não é permitido.

2.1.2 Interação com o Sistema e Envio de Mensagens

O sistema permite que vários utilizadores se conectem e interajam em simultâneo desde que cada um tenha a sua própria instância. Após estarem conectados, cada utilizador pode enviar e receber mensagens através dos tópicos aos quais está subscrito, ou deseja subscrever. As mensagens são identificadas com um prefixo, dependendo da origem:

- **Mensagem do manager**: A mensagem é precedida pela tag <MANAGER>.
- **Mensagem de um utilizador**: Um utilizador pode receber mensagens de outro(s) utilizador(es), se uma mensagem for enviada para um tópico comum entre eles. Quando isso acontece a mensagem é enviada contendo a “<Nome_Do_Topico> Nome do utilizador – mensagem enviada”.

O envio de mensagens para os tópicos tem as seguintes regras:

- Um utilizador pode enviar mensagens para **tópicos subscritos, tópicos que ainda não estejam criados ou tópicos que deseje subscrever**.
- Caso o tópico não exista no momento do envio, o **manager** cria-o automaticamente, desde que o número máximo de tópicos não tenha sido atingido. Os tópicos devem ser identificados por uma única palavra (por exemplo, "ISEC",

"informática", "cantina", etc.). Após a criação do tópico, o utilizador é subscrito ao tópico automaticamente e a sua mensagem é enviada. Caso o utilizador não esteja inscrito em determinado tópico e envie uma mensagem para o mesmo, a sua subscrição também é feita automaticamente e a sua mensagem é enviada.

- As mensagens enviadas serão automaticamente distribuídas para todos os utilizadores que estejam subscritos naquele determinado tópico.
- O utilizador, assim que desejar, pode deixar de subscrever um determinado tópico.

2.1.3 Tipos de Mensagens: Persistentes e Não-Persistentes

As mensagens enviadas pelos utilizadores podem ser de dois tipos:

- **Mensagens Não-Persistentes**: No momento do envio, o utilizador pode definir a duração da mensagem a zero. Essas mensagens são descartadas pelo manager após serem entregues a todos os utilizadores.
- **Mensagens Persistentes**: Ficam armazenadas pelo manager durante o tempo definido pelo utilizador. Durante este período, elas serão entregues a todos os utilizadores que se conectarem ao sistema e subscreverem o tópico em questão, mesmo que o sistema seja encerrado e reiniciado. Essas mensagens são armazenadas num ficheiro e são recuperadas automaticamente quando o manager é reiniciado.

2.1.4 Limitações do Sistema

O sistema possui algumas limitações, o que faz com que o funcionamento descrito anteriormente, só ocorre com sucesso quando elas não são excedidas. Essas limitações incluem:

- **Número máximo de utilizadores**: O sistema permite um máximo de **10 utilizadores** conectados simultaneamente.
- **Número máximo de tópicos**: O número de tópicos é limitado a **20**.
- **Número máximo de mensagens persistentes por tópico**: Cada tópico pode armazenar no máximo **5 mensagens persistentes**.

Em caso de tentativa de exceder uma dessas limitações, o sistema envia uma mensagem a informar que não é possível, explicando a causa.

2.1.5 Funcionalidade do Manager

Sendo o **manager** responsável por gerir todo o funcionamento do sistema, é ele que realiza as seguintes funções:

- **Criação e gestão de tópicos**: O manager cria tópicos automaticamente quando necessário, garante que o número máximo de tópicos não seja ultrapassado e remove-os no caso de um determinado tópico não conter nenhuma mensagem persistente ou nenhuma subscrição.
- **Armazenamento e distribuição de mensagens**: Ele lida com o armazenamento das mensagens persistentes em ficheiro, e trata da entrega dessas mensagens aos utilizadores subscritos nos tópicos dessas mensagens.
- **Controlo dos utilizadores**: O manager valida os utilizadores, garantindo que o número máximo de utilizadores não seja atingido, e distribui as mensagens conforme a subscrição dos utilizadores.

Além disso, através do manager administrador pode executar comandos para monitorizar e controlar o sistema. Por esse motivo e de modo a cumprir com os requisitos funcionais do projeto, o sistema **não permite** que existam múltiplos managers em execução simultaneamente, nem que os *feeds* permaneçam ativos após o manager se desconectar.

3 MECANISMOS DE SINCRONIZAÇÃO E COMUNICAÇÃO

A comunicação entre os componentes do sistema é feita através da utilização de **FIFOs** (Named Pipes), que são usados para a troca de mensagens entre o manager e os utilizadores através do programa *feed*. A comunicação também envolve o uso de **sinais** e **sincronização de threads** e **mutexes** para garantir que todos os processos sejam sincronizados corretamente e que as mensagens sejam enviadas e recebidas de forma eficiente.

- **FIFOs**: São utilizados no projeto para estabelecer a comunicação e garantir o envio e receção de comandos, mensagens e notificações de forma bidirecional entre o manager e os utilizadores. Assim, o sistema utiliza um FIFO para o manager (npServer) para envio e um FIFO exclusivo para cada feed (npCliente) que inclui o seu PID.
- **Sinais**: São utilizados para tratar eventos no sistema, tal como quando ocorre a saída de um utilizador (por exemplo, quando o utilizador pressiona Ctrl+C na linha de comandos).
- **Threads**: O sistema utiliza threads para permitir que o feed receba mensagens do manager enquanto o utilizador interage em simultâneo com a interface, tal como, o manager possa receber e validar logins a qualquer instante, enquanto o administrador interage com o manager por comados e o manager recebe e responde os pedidos(comandos) do manager. Com essa utilização, as operações ocorrem de forma mais eficiente e sem bloqueios.
  - **No Feed**:
    - **Thread listen_manager**: Esta thread é responsável por ficar em escuta pelas respostas do **Manager** em tempo real. Ela fica em espera, aguardando por mensagens do **Manager** e, assim que uma mensagem chega, a thread a processa e mostra ao utilizador.
  - **No Manager**:
    - **Thread manage_message_lifecycle**: Esta thread gere o ciclo de vida das mensagens, incluindo o envio, a persistência e a remoção de mensagens. Ela assegura que as mensagens sejam processadas de forma ordenada e que a sincronização entre os utilizadores seja mantida.
- Mutexes: Foram utilizados para evitar condições de concorrência e garantir mais segurança no acesso aos dados compartilhados. No sistema, eles são usados principalmente no Manager para garantir que as múltiplas threads que interagem com dados compartilhados, como mensagens, não acedam ou modifiquem os dados ao mesmo tempo de maneira inconsistente. Um dos exemplos da sua utilização no sistema é no manager_lifecycle.c, na função load_persistent_messages() para garantir que a leitura e manipulação de tópicos e mensagens persistentes seja feita de forma segura.

4 ESTRUTURA E ORGANIZAÇÃO DO PROJETO

O código do projeto é dividido em diversos ficheiros que implementam funcionalidades específicas para o *manager* e o *feed*. Além dos ficheiros principais (feed e manager), o projeto foi dividido para facilitar a organização do projeto e garantir melhor leitura e manutenção do código, bem como modularizar a execução de tarefas compartilhadas.

4.1 Ficheiros do Feed

- feed.h

Este ficheiro contém as declarações das funções que gerem a comunicação e interação do *feed* com o *manager*. Inclui funções para receber sempre as mensagens do *manager* e tratar sinais do sistema.

Função	Descrição da Função
listen_manager()	Fica em escuta e recebe mensagens enviadas pelo *manager*.
handler_sigint()	Trata dos sinais do feed e mandados pelo manager.

Tabela 1- Descrição feed.h

- feed_comandos.h

Aqui se encontra a declaração da função responsável pela validação dos comandos inseridos pelos utilizadores na interface do *feed*.

Função	Descrição da Função
validaComando()	Valida o comando inserido pelo utilizador para garantir que seja reconhecido pelo sistema.

Tabela 2- Descrição feed_comandos.h

4.2 Ficheiros do Manager

- manager_lifecycle.c / manager_lifecycle.h

Contém as funções que gerem o ciclo de vida do *manager* e consequentemente das mensagens persistentes.

Função	Descrição da Função
initialize_manager()	Inicializa o ciclo de vida do *manager*, e prepara os recursos necessários para o início do seu funcionamento.
finalize_manager()	Termina o ciclo de vida do *manager*, guardando as mensagens persistentes no ficheiro e terminando o mutex.
manage_message_lifecycle()	Gere o ciclo de vida das mensagens, incluindo o envio e a remoção de mensagens.

load_persistent_messages()	Carrega as mensagens persistentes do ficheiro para a memória ao iniciar o *manager*.
save_persistent_messages()	Guarda as mensagens persistentes no ficheiro para recuperação futura.

Tabela 3- Descrição manager_lifecycle

- manager_messages.c / manager_messages.h

Inclui as funções relacionadas à gestão das mensagens no sistema, como envio, armazenamento e recuperação de mensagens.

Função	Descrição da Função
send_message()	Envia uma mensagem para um *feed* específico.
get_username_by_pid()	Obtém o nome de utilizador através do PID a que está associado.
armazena_mensagem()	Armazena uma mensagem persistente para um tópico específico.
recupera_mensagem()	Recupera uma mensagem persistente do ficheiro e armazena na memória.
remove_message()	Remove uma mensagem de um tópico específico.
show_messages()	Mostra todas as mensagens persistentes de um tópico.
handle_message()	Trata o envio de mensagens para os utilizadores.

Tabela 4- Descriçãomanager_messages

- manager_topics.c / manager_topics.h

Aqui estão contidas as funções para a gestão e manipulação dos tópicos, tais como a adição, remoção, listagem e bloqueio/desbloqueio de tópicos.

Função	Descrição da Função
find_topic()	Pesquisa um tópico pelo nome e retorna a sua referência.
add_topic()	Adiciona um novo tópico ao sistema, se possível.
list_topics()	Lista todos os tópicos disponíveis no sistema.
toggle_topic_lock()	Bloqueia ou desbloqueia um tópico, impedindo ou permitindo novas mensagens.
handle_topics()	Lida com o comando topics do *feed*, e envia a lista de tópicos disponíveis, se houver.
send_topics()	Envia a lista de tópicos para o *feed*.
send_topic_messages()	Envia todas as mensagens persistentes de um tópico para um *feed*.
handle_subscribe()	Trata o pedido de subscrição de um tópico por parte de um utilizador.
handle_unsubscribe()	Trata o pedido de cancelamento de subscrição de um tópico por parte de um utilizador.

Tabela 5- Descrição manager_topics

- manager_users.c / manager_users.h

Trata das funções para a gestão dos utilizadores, incluindo login, saída, remoção e listagem de utilizadores ativos no sistema.

Função	Descrição da Função
add_user()	Adiciona um novo utilizador ao sistema.
handle_login()	Lida com o processo de login de um utilizador no sistema.
handle_exit()	Trata a saída de um utilizador do sistema, removendo-o dos utilizadores ativos.
list_users()	Mostra uma lista de utilizadores ativos no sistema.
remove_user()	Remove um utilizador do sistema e da lista de utilizadores ativos.
close_program()	Encerra o programa de forma segura e comunica aos utilizadores.

Tabela 6- Descrição manager_users

manager.c/manager.h

Ficheiro principal do *manager*, que contém as declarações gerais e as estruturas de dados utilizadas em todo o sistema, como utilizadores, tópicos e mensagens.

Estrutura de Dados	**Descrição da** Estrutura
Mensagem	Representa uma mensagem no sistema e inclui o conteúdo, autor e tempo de expiração.
Topic	Estrutura de um tópico, com o nome, estado de bloqueio e mensagens associadas.
User	A estrutura User representa um utilizador e contém o nome do utilizador, PID e tópicos subscritos.
tdados	Estrutura utilizada para gerir as threads e sinalização.

Tabela 7- Descrição manager

4.3 Ficheiro Comum

utils.c/utils.h

Neste ficheiro está contida uma função utilitária que é utilizada tanto pelo *manager* quanto pelo *feed*.

Função	Descrição da Função
organizaComando()	Organiza e formata o comando inserido pelo utilizador, removendo espaços e caracteres desnecessários.

Tabela 8- Descrição utils

5 ARQUITETURA DO SISTEMA

A arquitetura do sistema foi desenvolvida com o objetivo de garantir a simplicidade, modularidade e eficiência, utilizando uma comunicação assíncrona entre os componentes principais **Manager** e **Feed**, através dos FIFOs,e garantindo a integridade das funcionalidades e operações através da sincronização com **threads** e **mutexes.** A arquitetura do sistema, com enfâse nos componentes e ficheiros descritos no capítulo 4, pode ser representada pelo seguinte diagrama:

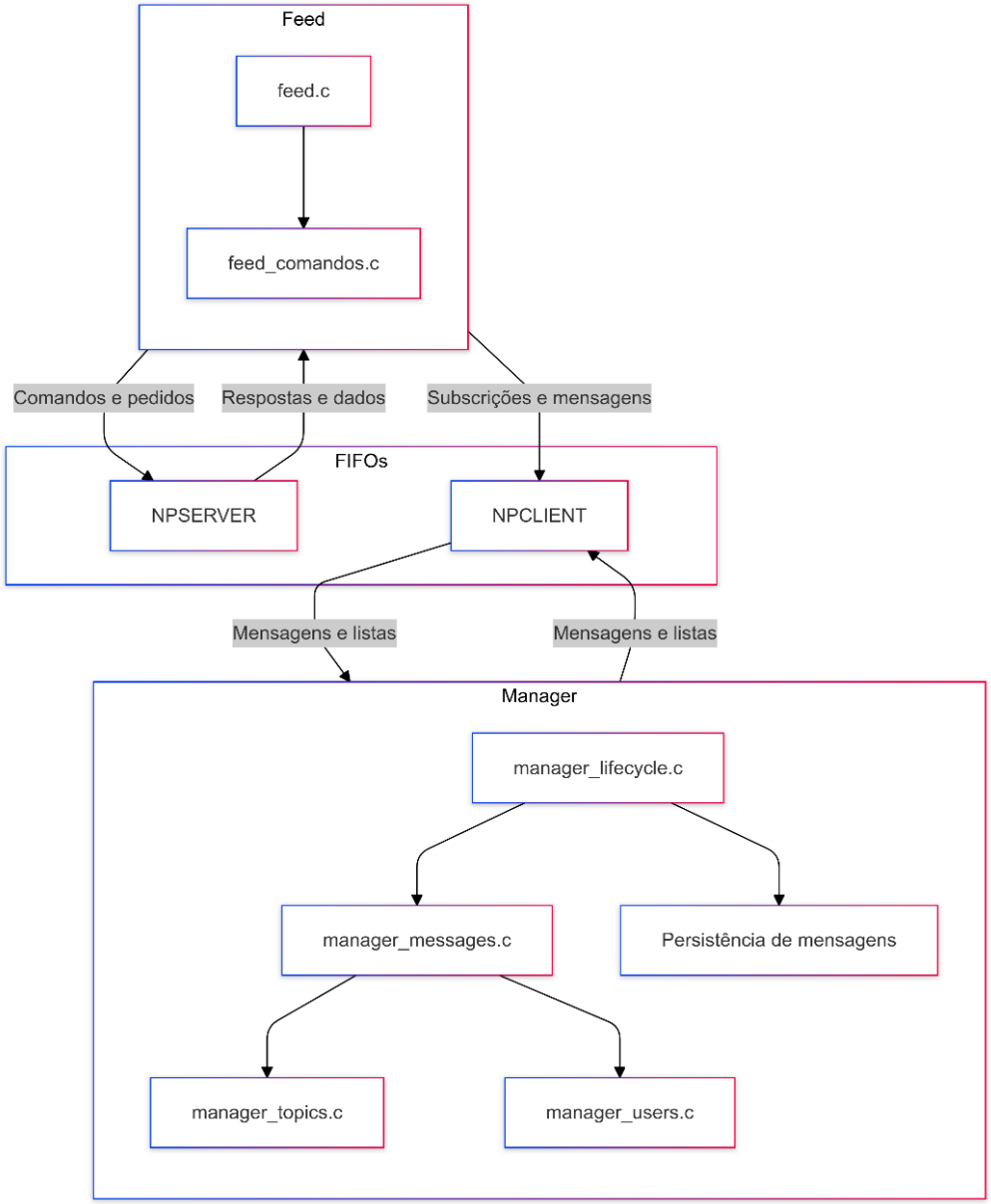

Figura 1- Diagrama da Arquitetura do Sistema

6 FUNCIONALIDADES IMPLEMENTADAS

6.1 Funcionalidades do Manager (Administrador)

Funcionalidade	Descrição	Estado
1. Listar utilizadores	O administrador pode listar os utilizadores atualmente a usar O sistema.	Implementado
2. Eliminar um utilizador	O administrador pode remover um utilizador dO sistema. O utilizador é notificado e o seu processo feed é terminado automaticamente.	Implementado
3. Listar tópicos existentes	O administrador pode listar os tópicos existentes, mostrando o nome e o número de mensagens persistentes de cada um.	Implementado
4. Listar mensagens de um tópico	O administrador pode listar as mensagens persistentes de um determinado tópico.	Implementado
5. Bloquear um tópico	O administrador pode bloquear o envio de novas mensagens para um tópico, mas as mensagens persistentes continuam a existir.	Implementado
6. Desbloquear um tópico	O administrador pode desbloquear um tópico, permitindo o envio de novas mensagens.	Implementado
7. Encerrar O sistema	O administrador pode encerrar O sistema, notificando os utilizadores e terminando todos os processos feed.	Implementado

Tabela 9- Funcionalidades Implementadas Manager

6.2 Funcionalidades do Feed (Cliente)

Funcionalidade	Descrição	Estado
1. Login	O feed inicia o programa com o seu nome de utilizador, que é validado pelo manager para permitir o acesso ou não ao programa.	Implementado
2. Verificar se o manager está a correr	O feed só pode ser executado se o manager estiver em funcionamento.	Implementado
Interação com o feed e sistema de feedback	**O utilizador consegue interagir com o feed utilizando comandos de texto e recebe feedback do manager para cada interação.**	Implementado
4. Comando topics	O utilizador pode listar todos os tópicos existentes, com o número de mensagens persistentes e o seu estado (bloqueado/desbloqueado).	Implementado
5. Enviar mensagem para um tópico	O utilizador pode enviar uma mensagem para um tópico, incluindo a duração da mensagem (persistente ou não).	Implementado
6. Subscrever a um tópico	O utilizador pode subscrever-se a um tópico e receber todas mensagens enviadas para esse tópico. Se existirem mensagens persistentes no momento da subscrição, o utilizador recebe-as.	Implementado
7. Deixar de subscrever a um tópico	O utilizador pode deixar de subscrever a um tópico, e deixa de receber as mensagens desse tópico.	Implementado
8. Comando exit	O utilizador pode sair do programa feed. O manager é notificado e o processo do feed é encerrando.	Implementado

Tabela 10- Funcionalidades Implementas Feed

7 CONCLUSÃO

O desenvolvimento deste sistema foi realizado de modo que fossem alcançados os objetivos propostos de forma eficiente e modular. A arquitetura adotada, com a separação entre o *manager* (servidor) e os *feeds* (clientes), e a utilização de mecanismos de concorrência e comunicação entre processos, como threads e FIFOs, garantiu que o sistema funcionasse com vários feeds em simultâneo e de modo leve.

A modularização e a separação do código facilitaram a análise e manutenção do sistema, proporcionando uma estrutura mais organizada. A utilização de boas práticas, como as de tratamento de erros, mensagens informativas para o **Feed** e para o **Manager**, e documentação e comentários explicativos foram chaves essenciais para a clareza e compreensão do código.

Durante o processo de implementação do sistema, foi possível explorar os conceitos essenciais aprendidos, o que proporcionou uma melhor compreensão das matérias lecionadas e do funcionamento de sistemas distribuídos em UNIX.

De forma geral, este trabalho proporcionou uma importante oportunidade de aplicarmos conceitos teóricos a um cenário prático, contribuindo para o aprimoramento das nossas habilidades em desenvolvimento de sistemas UNIX, programação em C e arquitetura de sistemas.